



HAL
open science

Un vecteur robotique polyvalent pour l'exploration sous-marine faible fond

Benoît Ropars

► **To cite this version:**

Benoît Ropars. Un vecteur robotique polyvalent pour l'exploration sous-marine faible fond. Micro et nanotechnologies/Microélectronique. Université Montpellier, 2015. Français. NNT : 2015MONTTS125 . tel-01980708

HAL Id: tel-01980708

<https://theses.hal.science/tel-01980708>

Submitted on 14 Jan 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

THÈSE

Pour obtenir le grade de
Docteur

Délivré par l'**université de Montpellier**

Préparée au sein de l'école doctorale I2S
Et de l'unité de recherche LIRMM

Spécialité : SYAM – Systèmes Automatiques et
Microélectroniques

Présentée par **Benoît ROPARS**

**Un vecteur robotique polyvalent pour
l'exploration sous-marine faible fond**

Soutenue le 16 décembre 2015 devant le jury composé de

M. David ANDREU, Maître de conférence HDR, UM
M. Lionel LAPIERRE, Maître de conférence, UM
M. Luc JAULIN, Professeur, ENSTA Bretagne
M. Kouider Nacer M'SIRDI, Professeur, AMU
M. Olivier PARODI, PHD, DCNS
M. René ZAPATA, Professeur, UM
M. David BOUCHAUD, CISCREA

Directeur
Co-encadrant
Rapporteur
Rapporteur
Examineur
Examineur
Invité



Remerciements

Je tiens à remercier tout particulièrement mon directeur de thèse M. David ANDREU et mon co-encadrant M. Lionel LAPIERRE pour leur pédagogie, leur soutien et leur patience, sans qui ce travail de thèse n'aurait pas abouti. Tous deux m'ont accordé leur confiance durant ces trois années de travail collaboratif. Un grand merci à vous deux.

Je remercie Pr. Kouider Nacer M'SIRDI et Pr. Luc JAULIN d'avoir accepté de rapporter cette thèse, ainsi que Pr. René ZAPATA et M. Olivier PARODI qui ont accepté le rôle d'examineur. Soyez assurés de mon entière reconnaissance.

Je tiens également à remercier l'équipe actuelle de Ciscrea, Claudy, Andreas, Xavier, mais tout particulièrement M. David BOUCHAUD pour m'avoir permis de réaliser cette thèse, un grand merci pour les moments passés dans vos murs au sein de votre équipe. Je fais aussi un petit clin d'œil aux anciens de Ciscrea qui ont toujours été là quand j'avais besoin d'un petit coup de main donc merci, Alexandre, Nadia et Silvère, pour votre amitié et votre soutien.

Merci également à l'équipe Explore du LIRMM (Didier, Sébastien, René, Pascal, Karen, Jean, Bruno, Lotfi, Yadpiroon, Nicolas, Florent et Robin) pour leur accueil, leur soutien et leur petits mots d'encouragement lors de la rédaction de ce manuscrit et tout au long de la thèse.

Je n'oublie pas de remercier toutes les personnes qui ont contribué à ce projet de près ou de loin. Je commence par mon colocataire de bureau Adrien avec qui j'ai partagé mes 3 ans de thèse sur le Jack. Il y a aussi Maxime et Silvain qui ont beaucoup fait pour ce projet, je n'oublie pas non plus les personnes que j'ai croisées dans le cadre de ce projet, Luc, PG, Yves et j'espère ne pas en oublier.

Enfin, je tiens à remercier également les personnes qui m'ont soutenu sur le plan personnel, ma famille et mes amis avec une pensée particulière pour Léa.

Remerciements	3
Introduction	9
Chapitre 2 : Le vecteur et ses applications	13
2.1 Une diversité d'applications	14
2.1.1 Inspection des voies navigables.....	14
2.1.2 Suivi et caractérisation d'écosystème	15
2.1.3 Inspection de zone archéologie sous-marine	19
2.1.4 Exploration et cartographie d'un réseau karstique	19
2.2 Un vecteur robotique polyvalent	26
2.2.1 Le vecteur	27
2.2.2 Morphologie du vecteur.....	28
2.2.3 Architecture matérielle (électronique).....	33
2.2.4 Architecture de commande.....	40
2.2.5 Architecture informatique.....	42
2.3 Positionnement – points clefs	42
Chapitre 3 : Abstraction et propriétés de l'étage d'actionnement : le répartiteur...45	45
3.1 Introduction.....	45
3.2 Notations	45
3.2.1 Modèle cinématique	45
3.2.2 Modèle dynamique	47
3.2.3 Simulateurs	49
3.3 Définition et réification de l'étage d'actionnement.....	50
3.4 Modélisation de l'étage d'actionnement	51
3.4.1 Modélisation d'un propulseur sous-marin.....	51
3.4.2 Modélisation de l'action de plusieurs propulseurs	52
3.4.3 Mise en œuvre expérimentale.....	54
3.5 Gestion de la redondance d'actionnement.....	61
3.5.1 Etat de l'art	61
3.5.2 Construction et analyse des propriétés du répartiteur.....	62
3.5.3 Exploitation de la redondance.	64
3.5.4 Compensation des zones mortes.....	64
3.5.5 Compensation des phénomènes dus à la disparité de caractéristique des moteurs.....	67
3.5.6 Généralisation au Jack 12	92
3.6 Points Clefs	96
Chapitre 4 : Architecture de Contrôle	99
4.1 Introduction.....	99
4.2 Modes de fonctionnement et ressources.....	101
4.3 Définition d'un mode de fonctionnement et d'un service.....	103
4.4 Définition de la configuration d'un service	104
4.5 Définition de la relation entre ressource, configuration et service.....	107
4.6 Définition de la relation entre service, configuration, ressource et « fonction ».....	109
4.7 Illustration du modèle d'architecture proposé, pour le robot Jack	111
4.7.1 Services et ressources DDL sur le robot Jack.....	111
4.7.2 Services et ressources capteurs sur le robot Jack	114
4.7.3 Configurations et fonctions	115
4.7.4 Changement automatique de configuration	115
4.8 Le middleware ContrAct.....	117
4.9 Projection de l'architecture sur le middleware ContrACT.	119
4.9.1 Structuration de l'architecture globale sur ContrAct.....	120
4.9.2 La gestion de service : descriptions basées XML.....	123
4.9.3 La gestion de service : liaisons basées sur les ports	128
4.9.4 La gestion de service : changement de mode	132

4.9.5	La gestion de service : exécution sous contrôle de l'ordonnanceur	137
4.9.6	Impact des contraintes de précedence non respectées	141
4.10	Points clefs	146
Chapitre 5	: Expérimentations.....	149
5.1	Les Expérimentations en piscines.....	149
5.2	Les Expérimentations terrains.....	151
5.2.1	Immersion du Jack 12 en environnement réel.	151
5.2.2	Inspection de structures immergées.....	153
5.2.3	Inspection de voie navigable	155
5.2.4	Suivi d'écosystème	158
5.2.5	Archéologie	160
5.3	Expérimentations à venir	161
5.3.1	Mayotte, novembre/décembre 2015	161
5.4	Points clefs	161
Conclusion et Perspectives	163
Bibliographie	169
ANNEXE I : Modules et Services	177
ANNEXE II : Architecture matérielle embarquée	185

Introduction

La robotique sous-marine, à l'image de la robotique d'exploration au sens général du terme, a connu d'importantes évolutions pour répondre à la diversité des enjeux qu'ils soient sociétaux, environnementaux, économiques, militaires et scientifiques. Les robots sous-marins qui ont été développés à travers le siècle écoulé sont des vecteurs permettant d'aller au delà des limites humaines afin d'acquérir des données, voire de manipuler des dispositifs, au sein d'environnements de grands fonds tels les mers et les océans. Les applications sont aujourd'hui nombreuses, avec une part importante d'applications relevant de l'industrie pétrolière, des télécommunications ou de l'étude des milieux océaniques. Ces vecteurs sont fréquemment, voire essentiellement, des robots de taille conséquente à l'image de la flotte de véhicules d'exploration de l'IFREMER tels le Victor 6000 et l'AsterX [1]–[3], le Romeo du CNR-ISSIA [4], les Hyper-dolphin et Kaiko7000II du JAMSTEC [5], [6] ou encore le Verdana et le Tiburon du MBARI [7], [8]. Les véhicules utilisés dans le milieu pétrolier sont également très imposants comme par exemple ceux de la compagnie mexicaine de recherche pétrolière PEMEX [9], [10] ; ces vecteurs permettent de réaliser des travaux en profondeur sur les puits de pétrole ainsi que de faire des inspections de pipelines [11]. Qu'il s'agisse de vecteurs de type ROV (Remotely Operated Vehicles), pilotés de la surface via un "cordon ombilical", ou de type AUV (Autonomous Underwater Vehicles), autonomes en énergie et décision, leur dimension et leur charge utile permet d'embarquer de nombreux capteurs et les moyens requis par l'accroissement de leur autonomie opérationnelle (qu'il s'agisse de la capacité à réaliser des tâches ou à gérer une mission).



Figure 1.1 : Le verdana du MBARI[8] (gauche) et le KAIKO7000 du JAMSTEC [6] (droite)

A contrario, encore bien peu de travaux et de vecteurs abordent l'exploration sub-aquatique, au sens de l'environnement aquatique faible fond¹ ou confiné tels que les lacs, les rivières, les réseaux karstiques, les zones portuaires, etc. C'est ce domaine d'application qui est considéré dans nos travaux. Il ne s'agit donc pas de robotique sous-marine dédiée à l'exploration des grands fonds marins ou océaniques ; une exploration exploitant des robots allant de plus de 100 kg à plusieurs tonnes et nécessitant une logistique conséquente. A l'opposé, ou plutôt en complément, nous visons une technologie légère, facilement déployable, adéquate pour l'exploration ou l'inspection en environnement aquatique, d'accès

¹ Pouvant atteindre plusieurs centaines de mètres, mais dans des contextes confinés comme une cavité karstique par exemple.

souvent difficile. En effet, ce domaine d'application, qui impose une logistique légère, nécessite de recourir à des engins de faible encombrement et offrant une grande manœuvrabilité, comme en témoigne la diversité d'exemples d'applications illustrant tant le besoin que l'intérêt d'une telle technologie : l'exploration de sources d'eau douce et des réseaux karstiques, le suivi de la qualité des eaux ou l'étude du fonctionnement des écosystèmes lagunaires, l'inspection de canaux d'irrigation ou de voies navigables, la recherche archéologique subaquatique en rivière ou lac, l'inspection de structures immergées tels que les barrages, ou encore l'inspection d'aménagements portuaires (lignes de mouillage, récifs artificiels), etc. Ces exemples d'applications seront par la suite considérés comme les « applications métiers ».

Dans ces différents exemples un système robotisé d'inspection, plus ou moins autonome, revêt un intérêt évident et la charge utile, les capteurs embarqués principalement, est souvent différente. Vu sous l'angle applicatif, le robot est souvent considéré comme un porteur de capteurs. Néanmoins, baser les stratégies d'exploration, et donc de contrôle, sur les exigences métiers et les propriétés de l'environnement rend clairement cette vue réductrice ; il s'agit plutôt de capteurs à mobilité contrôlée, cette dernière étant potentiellement basée sur l'environnement et/ou la stratégie métier.

Vu sous l'angle métier, i.e. celui des utilisateurs, cela contribue à :

- L'enrichissement des données acquises, par la complémentarité entre les capteurs environnementaux (la charge utile des utilisateurs, par exemple : sonars, caméras, CTD², fluorimètres et autres capteurs divers) et ceux intrinsèquement nécessaires au contrôle du vecteur (centrale inertielle, loch doppler, etc.) et à la communication.
- La possibilité de positionner les capteurs dans l'espace (milieu aquatique), voire de conduire les capteurs selon une trajectoire désirée (espace et temps) ; une mobilité contrôlée qui permet différentes stratégies d'exploration du milieu ou d'inspection de la cible.
- L'enrichissement des données acquises et contrôle de la mobilité, potentiellement en réseau (flottille), permettent d'affiner ou de construire des modèles de l'environnement, donc une image plus riche (fusion ou corrélation des données) du milieu ou de la cible.

Nos travaux, réalisés dans le cadre d'un partenariat industriel (Ciscrea³), prennent également en considération la dimension économique, à travers l'émergence de produits et de services construits sur un même « produit de base ». Les exemples que nous avons mentionnés laissent effectivement apparaître le besoin d'une technologie évolutive permettant de répondre à la diversité des applications, et donc des services associés. Nos travaux visent donc en premier lieu la conception et la mise au point d'une technologie adéquate et évolutive, au sens configurable pour s'adapter aux exigences des différents exemples d'applications métiers.

Le propos mentionne volontairement la notion de services au sens de ce que la technologie doit permettre dans l'acquisition de données pour enrichir la connaissance, pour produire des modèles et des informations permettant de comprendre, voire de prédire, le fonctionnement de systèmes ou d'écosystèmes liés à l'environnement et, par effet de levier, pour développer/renforcer les compétences de Ciscrea et de ses clients (i.e. des utilisateurs de ses robots).

² Capteur de Conductivité-Température-Pression

³ www.ciscrea.net

En termes de technologie, nos travaux s'inscrivent dans la création de vecteurs robotisés légers (30-50kg), de petit volume (environ 0.3 m³). Ces vecteurs robotisés doivent être facilement déployables, et donc ne pas nécessiter de moyens logistiques lourds, et doivent pouvoir être déclinés sous forme de ROV ou d'AUV selon les cas.

Le terme de « technologie » couvre aussi, plus largement, les aspects relatifs aux modèles et aux données, et surtout à leur traitement (logiciel) pour les classifier et en extraire les informations pertinentes et utiles sur un plan « métier ». Néanmoins, nous nous concentrerons sur la technologie au sens robotique et sur les défis que représente la création d'un vecteur robotique adaptable à ces différents besoins applicatifs : créer un « **vecteur robotique polyvalent pour l'exploration sous-marine faible fond** ». Il s'agit en d'autres termes dans cette thèse CIFRE de concevoir, à partir du robot existant mis au point par Ciscree, un robot sous-marin que l'on peut faire évoluer aisément tant sur le plan mécanique, électronique qu'informatique. Ce dernier point sous-entend les algorithmes de contrôle/commande embarqués dans le véhicule, offrant différents modes de fonctionnement tels que les modes téléopérés et autonomes (plusieurs niveaux de téléopération et d'autonomie pouvant être envisagés). De cette étude une gamme de robots doit être déclinable, d'une version de base où le contrôle du véhicule est intégralement confié à l'opérateur à un véhicule capable d'évoluer de manière autonome dans le cadre de la mission qui lui sera confiée. De même, cette gamme de robots peut reposer sur des véhicules légèrement différents en termes de charge utile (capteurs à embarquer) et point essentiel, de capacité d'actionnement.

Ces objectifs se déclinent en un ensemble d'aspects scientifiques et/ou technologiques à considérer parmi lesquels :

- L'évolution de la charge utile embarquée : selon le type de capteurs à embarquer il est parfois nécessaire d'ajouter un élément mécanique au robot, un châssis (appelé un « skid ») sur lequel il est possible de monter les capteurs encombrants comme par exemple les capteurs sonars pénétrateur de sédiments, bathymétrique, courantométrique, etc. L'ajout d'un skid, en dehors de considérations d'assemblage mécanique, modifie les propriétés hydrodynamiques du robot, et donc les modèles manipulés par le contrôle, et peut dans certains cas nécessiter l'ajout de puissance d'actionnement. L'ajout d'actionneurs suppose inévitablement sa prise en compte dans le contrôle, qui doit pouvoir exploiter la redondance structurelle du système, pour un gain de précision, de réactivité et de robustesse.
- L'exploitation des données capteurs : au delà de la connectique relative aux capteurs embarqués, se posent des problèmes de recueil et d'exploitation de ces données capteurs. Le recueil peut être problématique dès lors que différents capteurs sonars (i.e. basés sur des ultrasons) sont utilisés. En effet, ces capteurs peuvent être interférents et donc leur utilisation peut nécessiter une stratégie de recrutement des capteurs, situation qui est rarement rencontrée en robotique terrestre. L'orchestration du recrutement des éléments potentiellement interférents appuie la question de l'architecture logicielle qui, en particulier dans un contexte de flottille, reste une question ouverte.
- L'offre de différents modes de contrôle du robot : selon le besoin et donc le modèle dans la gamme de robots proposée, différents modes de fonctionnement doivent être possibles, de la télé-opération directe (bas niveau) à la télé-opération haut niveau (potentiellement télé-programmation), voire même à un fonctionnement autonome. Ces différents modes, ainsi que les différentes applications métiers envisageables, conduisent nécessairement à

proposer un ensemble de modalités de commande et de supervision, en charge notamment de l'adaptation de la commande au contexte. Cette question d'une autonomie variable, adaptative est centrale dans les problématiques actuelles de la robotique.

- L'évolution de l'électronique embarquée : les différents modes évoqués peuvent requérir l'ajout de ressources électroniques (carte processeur par exemple) pour supporter les calculs que les modes évolués nécessitent. Le partitionnement des « fonctionnalités » tant sur le plan de l'architecture logicielle de contrôle que de l'architecture matérielle est essentiel pour répondre à la propriété d'évolutivité.

Cet ensemble d'aspects à considérer a un impact évident sur l'architecture, tant matérielle que logicielle, et impose de proprement conceptualiser cette « polyvalence », en s'attachant à apporter l'abstraction requise et à réifier dans l'architecture de contrôle tout ce qui doit être explicitement « manipulable ».

Le manuscrit se compose de 5 chapitres. Le chapitre 2 décrit de façon plus précise le contexte applicatif, en détaillant diverses applications et en mentionnant quelques vecteurs existants. Les exigences que la polyvalence implique sur la mécanique, l'électronique, l'informatique et l'automatique, sont présentées, considérant les préoccupations économiques de l'industriel. La première partie du chapitre 3 est consacrée à la notation qui sera utilisée dans le document. Les différents modèles cinématiques et dynamiques y sont introduits. Le simulateur qui est mis en œuvre dans le chapitre suivant est présenté et la seconde partie aborde l'expression de la polyvalence centrée sur la question de l'actionnement. Nous introduisons une abstraction de l'étage d'actionnement et en étudions les propriétés que sa redondance lui confère. Le chapitre 4 poursuit la réflexion sur la polyvalence mais du point de vue de l'informatique. Le chapitre 5 expose les expérimentations qui ont été conduites soit en piscine soit sur le terrain, afin de tester les développements réalisés pour concrétiser nos propositions.

Ce manuscrit s'achève par une conclusion suivie d'une discussion sur les perspectives ouvertes à l'issue de ces travaux.

Chapitre 2 : Le vecteur et ses applications

Le domaine applicatif de nos travaux est l'exploration robotisée de l'*environnement aquatique faible fond ou confiné*. Cet environnement désigne les eaux « terrestres » au sens des lacs, rivières, karsts, gouffres, cénotes, barrages, etc. ainsi que les eaux faible fond se situant en zone côtière telles que les lagunes, les ports, etc. au sein desquelles l'accès (humain) est potentiellement difficile, voire dangereux (karst), ou nécessite de petits vecteurs et une logistique légère. Le vecteur doit permettre de conduire un ensemble de capteurs au sein du milieu pour contribuer à son inspection, sa caractérisation, son suivi, etc.

Nous allons évoquer la diversité d'applications par quelques exemples non exhaustifs. Le but de cette illustration est de montrer l'importance de la polyvalence du vecteur et ses limitations actuelles. En effet, conçu pour être polyvalent, le vecteur doit pouvoir être adapté tant dans sa morphologie – architecture mécanique permettant de s'adapter à la charge utile (capteurs) – que dans son architecture embarquée – architectures électronique et informatique (intelligence embarquée). Nous mentionnerons au fil des exemples l'adaptation nécessaire du vecteur à chaque changement de la charge utile dictée par l'application. Le propos est de souligner la nécessité de la **polyvalence** sur le plan technologique, en réponse à des préoccupations de stratégie d'entreprise visant la conception d'une gamme de robots et de services associés.

Le vecteur mentionné dans ces différentes applications est un prototype de vecteur développé au LIRMM, basé sur le robot Jack commercialisé par l'entreprise CISCREA⁴ (Figure 2-1).



Figure 2-1 : Le robot Jack de la société CISCREA

⁴ www.ciscrea.net

2.1 Une diversité d'applications

2.1.1 Inspection des voies navigables

Un des intérêts d'une inspection des voies navigables est d'inspecter l'état des voies pour leur entretien (détection d'objets encombrants, comme des véhicules par ex.), pour mesurer l'érosion des berges, pour vérifier l'état de structures telles que des écluses, etc [12]. Si un contrôle visuel par caméra sous-marine est parfois possible quand l'eau n'est pas trop turbide, il faut néanmoins rester proche des parois à observer. Il est dès lors difficilement envisageable de parcourir de grands volumes et le recours à une image acoustique offre non seulement des données plus précises mais offre un gain de temps considérable (l'acquisition pouvant être réalisée à plusieurs dizaines de mètres). Pour ce faire, il est nécessaire de monter sur le vecteur sous-marin un sonar profilométrique afin de réaliser par exemple l'acquisition de données requise par la reconstruction du modèle 3D des berges. Le contrôle de l'engin devra être également pris en compte, car celui-ci est impacté par les modifications apportées au vecteur. Cet exemple paraît simple, et il l'est si le vecteur est directement piloté par l'homme via un cordon ombilical (lien physique reliant le poste opérateur au vecteur), ce qui est possible dans cette application. Par contre, automatiser le pilotage du vecteur, à savoir conduire le capteur selon une trajectoire définie, est plus compliqué dès lors que le contrôle doit se baser sur les données acoustiques acquises, comme cela est le cas pour le suivi automatique de berge, de paroi, etc. Pour réaliser cette application, il est nécessaire de doter le robot d'un « skid » (berceau) pour pouvoir embarquer un sonar profilométrique classique, i.e. commercialisé pour les robots sous-marins, qui est habituellement de taille, de volume et de poids conséquents⁵. La consommation est également un facteur important à considérer.

De nombreux vecteurs d'inspection pilotée par l'homme sont disponibles. Sans être exhaustif, nous pouvons mentionner ceux proposés par VideoRay⁶, Ac-cess⁷ ou Subsea tech⁸. Une des limitations de ces vecteurs vient de leur faible capacité d'emport de capteurs et de leur sous-actionnement qui ne leur permet pas de réaliser l'ensemble des mouvements nécessaires à la réalisation d'une mission telle que l'inspection automatisée d'une voie navigable. Illustrons ces propos en considérant le robot Ac-rov de Ac-cess et l'Observer de Subsea-tech (Figure 2-2).



a) Ac-Rov de Ac-cess



b) Observer de Subsea-tech



c) Videoray Pro 4 de Videoray

Figure 2-2 : Exemple de Mini ROV

⁵ A titre d'exemple, le sonar que nous utilisons est un sonar profilométrique Super SeaKing Profiler de Trittech, d'une longueur de 300 mm et de diamètre 110 mm, pour un poids de 3.5kg en air et de 1.7kg en eau.

⁶ www.videoray.com

⁷ www.ac-cess.com

⁸ www.subsea-tech.com

Le manque de capacité d'emport de ces robots vient de leurs petites dimensions (203 x 152 x 146 mm pour l'Ac-rov et 490 x 270 x 210 mm pour l'Observer). Ces robots sont vraiment orientés pour de l'inspection visuelle et ne disposent pas d'un contrôle avancé, seul les suivis de consignes en cap et en profondeur sont parfois disponibles sur ce type de vecteurs faisant partie de la classe des mini ROV. Il est inimaginable de transporter de très gros capteurs qui parfois sont plus lourds que le robot lui-même. Ce type de robot n'est généralement pas autonome en énergie, seuls les robots de subsea-Tech cités plus haut embarquent leurs batteries.

La motorisation est également un point clé pour les ROVs de petites tailles amenés à être exploités dans des conditions où leurs faibles encombrements et leurs facilités de mouvements sont essentiels. Très peu de mini-ROVs cités dans cet état de l'art contrôlent plus de 4 à 5 degrés de libertés, excepté, dans la gamme supérieure, le vecteur LBV-10 de Seabotix⁹ qui lui contrôle les 6 degrés de libertés (Figure 2-3).

Par gamme supérieure nous entendons les vecteurs de taille plus conséquente que les mini-ROV, permettant d'embarquer une charge utile plus importante. Les vecteurs commercialisés par Ageotec¹⁰, Rovtec¹¹, SAAB¹² ou Seabotix¹³ relèvent de ce type de ROV.

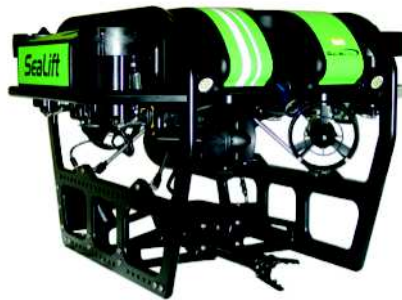


Figure 2-3 : ROV LBV-10 de Seabotix

Tous ces vecteurs sont dédiés à l'exploration des milieux sous-marins, et dans la majorité des cas, sont spécialisés pour une ou deux tâches (au sens applicatif) telles la bathymétrie ou l'inspection de structure. Leur conception initiale impose des modifications structurelles sur l'aspect mécanique, ainsi que sur l'aspect logiciel, pour modifier ou ajouter une charge utile (i.e., pour s'adapter à d'autres applications).

2.1.2 Suivi et caractérisation d'écosystème

L'inspection et la caractérisation de l'évolution de la faune et de la flore de récifs artificiels sont un exemple d'application permettant d'évaluer la restauration/compensation écologique [13]–[15]. Il est loin d'être trivial selon la localisation des récifs – au sein d'un port sous des pontons et des bateaux, ou dans un espace non contraint – et selon la nature de la

⁹ www.seabotix.com

¹⁰ www.ageotec.com

¹¹ www.rovtechsystems.co.uk

¹² www.seaeye.com

¹³ www.seabotix.com

caractérisation, qualitative – seulement une inspection visuelle – ou quantitative – avec des mesures objectives telles que le nombre et le type de poissons, la croissance des algues, etc. Cette inspection permet de visualiser l'état de développement d'un récif artificiel. On peut également réaliser cette inspection dans le cadre d'un milieu naturel afin de caractériser l'évolution de la faune et de la flore des récifs naturels. Cette inspection permet d'alimenter les études sur la biodiversité marine, notamment dans le cadre de l'étude des effets des changements locaux et globaux liés à l'anthropisation sur les écosystèmes lagunaires et marins côtiers. Cette application permet de visualiser l'état d'un récif naturel, d'effectuer différents types d'observations dont suivre des transects (chemin référencé au terrain), de faire des observations localisées (tourner autour d'une zone d'intérêt avec une attitude d'observation désirée et contrôlée) et suivre des espèces (composition des 2 modes précédents). Pour se faire, le vecteur nécessite l'emport d'une caméra haute définition et d'une caméra acoustique afin de réaliser une acquisition de données et un contrôle de la plateforme. Dans le cas d'un suivi d'écosystème dans un milieu naturel, un capteur supplémentaire permettant de connaître l'altitude de l'engin et un contrôle avancé sont nécessaires. L'objectif n'est pas d'avoir des contrôles compliqués quand un simple pilotage direct est possible. Mais piloter directement tous les degrés de liberté d'un vecteur sous-marin et ce en présence d'obstacles (par ex. pontons, coques de bateaux) et de perturbations (courants marins, houle), tout en assurant un retour vidéo stable n'est pas si trivial. Un ensemble de fonctions de base doivent être embarquées pour alléger la charge opérateur, comme les contrôles en profondeur et en cap par exemple. S'il ne s'agit pas d'une simple inspection visuelle, il faut stabiliser le vecteur pour maintenir le capteur vidéo sur un point précis de l'espace, il est alors nécessaire d'embarquer soit 2 caméras (stéréovision), soit un pointeur laser (reconstruction de la profondeur de l'image), soit avoir recours à un échosondeur. Exploiter en ligne ces données dans le cadre du contrôle du vecteur suppose alors de les traiter, de construire les modèles simplifiés et ce via des algorithmes performants (pour une exploitation en ligne). Pour un contrôle avancé, il sera nécessaire d'ajouter un caisson dédié, contenant un processeur capable de traiter des contrôles complexes nécessitant des modèles de commande avancés et des traitements bien plus lourds. Un autre point à mentionner concernant cette application, est le cordon ombilical. Le retour vidéo n'est possible que grâce au lien ombilical via lequel l'opérateur pilote le vecteur. Si de multiples données doivent être retournées, il faut passer d'une solution cuivre à une solution fibre optique. Mais au delà de cette caractéristique, la présence du câble perturbe le pilotage du robot et induit un risque de blocage du vecteur dans l'environnement ; le risque est ici maîtrisé car l'accès est facile, mais cela est un réel défi pour l'exploration de réseaux karstiques évoqués par la suite.

Les robots utilisés dans le suivi d'éco-système sont souvent très lourds à mettre en œuvre et le prix d'un tel déploiement est souvent élevé (tant sur le plan matériel qu'humain). En France, l'Ifremer est un acteur majeur pour l'étude et la caractérisation de la biodiversité marine (axe stratégique 2¹⁴). Il dispose d'une flotte de robots sous-marins pour l'exploration scientifique des océans. Des robots comme le Victor est un ROV 6000 mètres de 4.6 tonnes avec comme dimensions 3.1 x 1.8 x 2.1m [2]. La mise en œuvre de ce robot nécessite l'utilisation de gros navires (L'Atalante, Le Thalassa ou Le Pourquoi pas?) disposant de moyens de levage suffisant pour supporter le poids de celui-ci. Les actions de l'Ifremer se concentrent surtout sur de l'exploration grand fond, ce qui explique les dimensions des engins qu'il opère.

¹⁴ <http://wwz.ifremer.fr/Les-sciences-marines/Axes-strategiques>



Figure 2-4 : ROV Victor de l'Ifremer (4.6 tonnes)[16]

Des vecteurs plus légers sont également utilisés comme le Phantom S2 et le Phantom 300 de l'Université de Caroline du Nord (USA) qui permettent de faire de l'étude des coraux [17]. La gamme des engins Phantom affiche un poids de 135 Kg pour une charge utile de 14.8 Kg. Les performances remarquables de cet engin se situent surtout au niveau de son autonomie énergétique. Il peut réaliser des missions d'une durée de plus de 10h. La conséquence est bien sûr son poids résultant qui dépasse les 100 Kg.

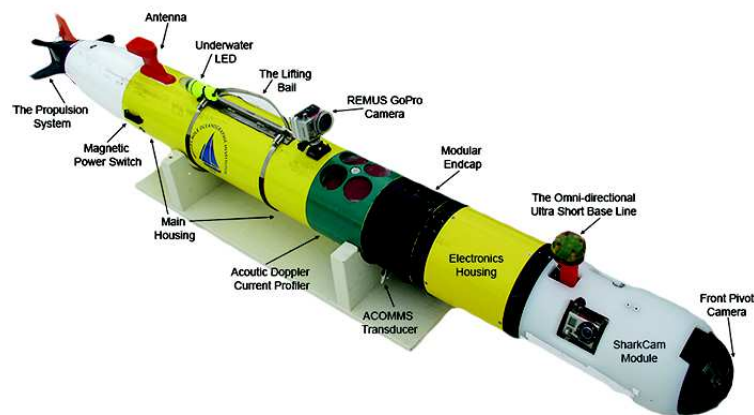


Figure 2-5 : Le robot REMUS équipé de la "SharkCam"[18]

Dans le suivi d'espèce, des expérimentations ont été réalisées sur le suivi de requin [18] et de tortues [19], cette étude a été réalisée par le WHOI (Woods Hole Oceanographic Institute) à l'aide d'un sous-marin Remus de l'industriel Kongsberg¹⁵. Le Remus est un engin de type torpille, sous-actionné, qui est mis en œuvre dans de nombreux contextes applicatifs, scientifique, sécurité civile ou militaire. Son caractère sous-actionné le destine à des missions au long cours qui sortent du contexte de notre étude. Il faut aussi mentionner dans la même

¹⁵ www.km.kongsberg.com

gamme l'engin Gavia, développé par Teledyne¹⁶ ainsi que l'AUV MARES développé par l'université de Porto au Portugal [20]. Tous deux s'attachent à prendre un soin particulier au design modulaire de l'engin comme dans les travaux de [21] qui traite de cette modularité.



Figure 2-6 : Le MARES de l'université de Porto (Portugal) [20]

Dans le contexte du suivi d'écosystèmes, une question intéressante est de connaître les effets de la technologie sur les habitats naturels ; c'est dans ce cadre que l'Université de Washington a développé un système de capteurs (Monitoring Package) permettant de connaître les effets des centrales marémotrices sur le comportement des poissons. Nous ne décrirons pas ce que contient le capteur, si ce n'est pour souligner les volume et masse importants qu'il représente. Pour embarquer un tel capteur la motorisation du ROV «Millennium Falcon»[22]–[24] a été modifiée (Figure 2-7), avec une approche similaire à celle que nous proposons plus loin.



Figure 2-7 : Le "Millennium falcon"[22]

¹⁶ www.teledynegavia.com

2.1.3 Inspection de zone archéologie sous-marine

Un des intérêts d'une inspection des zones archéologiques est de proposer aux archéologues de pouvoir accéder à des zones pour l'instant jamais explorées. Le vecteur permet d'identifier des zones intéressantes en visualisant des objets posés sur le fond, comme nous l'avons constaté lors de notre mission archéologique en Bolivie, à l'aide du Jack, destinée à de la prospection visuelle avant l'intervention des plongeurs archéologues. Cette mission est réalisée dans le cadre de recherche sur la civilisation Tiwanaku (projet Huiñamarca), financée par l'Université Libre de Bruxelles (Belgique) [25]. Le fait d'utiliser un ROV léger permet de faciliter la mise en œuvre par une petite équipe, élargissant les zones de prospection. Néanmoins, équiper le vecteur sous-marin d'une charge utile de type sonar sédimentaire permettrait d'apporter une contribution plus importante en détectant d'éventuels objets enfouis dans les sédiments. A l'heure actuelle, les missions d'inspection du sédiment à l'aide de ce type de capteur sont réalisées à l'aide de systèmes remorqués par des bateaux, comme le propose EdgeTech¹⁷.

Dans certains cas le vecteur peut également être amené à collecter des objets sur le fond. Cependant, pour éviter de les briser, des recherches sont réalisées sur une nouvelle technologie de pince (main anthropomorphe) ainsi que son contrôle. C'est l'objectif du robot « Speedy » développé au LIRMM dans le cadre de recherches archéologiques portant sur des vestiges du 17^{ème} siècle de l'épave de la Lune [26].



Figure 2-8: Le miniROV Speedy développé au Lirmm [26]

2.1.4 Exploration et cartographie d'un réseau karstique

Cette application représente un enjeu sociétal majeur. La quantification des réserves d'eau douce exploitables dans les bassins karstiques (sur tout le bassin méditerranéen) est une question essentielle qui conditionne les politiques publiques en rapport avec la gestion et la préservation de la ressource. De plus, les épisodes pluviométriques extrêmes que connaissent certaines régions excitent la dynamique des réseaux karstiques (peu connus), et leur décharge provoque de soudaines et dramatiques montées des eaux.

¹⁷ www.edgetech.com/products/sub-bottom-profiling/3100-portable-sub-bottom-profiler/

Nos partenaires hydrogéologues (académiques et industriels) ont développé une expertise sur ces questions mais manquent actuellement d'outils d'exploration profonde de manière à affiner la caractérisation des réseaux karstiques. Ainsi, l'objectif est de fournir des modèles terrain fiables permettant de prévoir la dynamique de décharge des réseaux karstiques et d'éclairer les décideurs sur un mode de gestion durable de la ressource. Pour prendre ce type de décision, il est important de dresser le modèle 3D du réseau karstique et d'effectuer des relevés physico-chimiques de la qualité de l'eau.

De multiples tentatives d'explorations ont eu lieu dans différents sites. Parmi les plus connues, en France, se trouvent Font Estramar et Fontaine de Vaucluse, auxquelles nous nous référons à plusieurs titres. Leur importance patrimoniale, écologique et sociétale est avérée.

Font Estramar serait, d'après l'hydrogéologue Henri Salvayre, la plus grande réserve d'eau douce du Sud de la France et peut-être même d'Europe [27][28]. La superficie de cette « mer souterraine » est estimée à 1000 km² pour une profondeur possible de 400 à 600 m. Des explorations humaines périlleuses ont malheureusement parfois conduit à des décès, mais également à un exploit réalisé en juillet 2015 avec une descente jusqu'à la profondeur de 262 m (le précédent record dans ce même gouffre était de 248m [29]).

Fontaine de Vaucluse est retenue ici au titre de la riche expérience de plongées de prospection robotisées ; elles ont certes abouti en 1989 par une prospection atteignant une profondeur de 315 m (robot Spélénaute) mais de nombreux vecteurs n'en sont pas revenus.

Acquérir cette connaissance sur l'environnement impose d'amener le robot au delà des limites humaines, celles des plongeurs spéléologues ; une application qui pose des défis scientifiques et technologiques importants.

2.1.4.1 L'exploration karstique robotisée de Fontaine de Vaucluse

Les éléments évoqués dans ce qui suit sont tirés des travaux de R. Pastor, de la Société spéléologique de Fontaine de Vaucluse¹⁸. En France, depuis le début des années 80, les ROV sont utilisés en spéléologie notamment à la Fontaine de Vaucluse [30]. Ce gouffre a été utilisé par J.Y Cousteau et F. Dumas afin de réaliser les premières plongées en scaphandre autonome. L'un des premiers engins télé-opérés fut le Télénaute de l'Institut de la Recherche Pétrolière en 1967 [30]. L'engin fut suivi par A. Falco jusqu'à la profondeur de -90 mètres, le robot, quant à lui atteint la profondeur de 106 mètres. Les premières missions de ces robots étaient surtout consacrées à l'exploration et à l'enregistrement vidéo. Début des années 80, le Télénaute fut remplacé par le Sorgonaute (Figure 2-9), décliné ensuite en quatre versions. Ce nouvel engin a été développé en collaboration avec l'équipe mécanique du constructeur automobile Renault [30]. Le premier des « Sorgonaute » a été mis à l'eau en janvier 1983. Cette première mission fut un réel succès mais malheureusement, il fut contraint d'y mettre un terme par manque de câble après une descente à -243 m.

¹⁸ <http://www.plongeesout.com/sites/provence/vaucluse>



Figure 2-9 : Sorgonaute (1983) [31]

Le 22 septembre 1984, le Sorgonaute II a été mis à l'eau mais fut perdu à cause d'un fil d'Ariane pris dans les hélices à environ -233 m de la surface. La profondeur maximale dans le gouffre (-315 m) fut réalisée par le Modexa (Figure 2-10) de la société marseillaise MIC en 1985.



Figure 2-10 : Modexa (1985) [32]

Ce ROV était initialement utilisé pour la recherche d'épave en Méditerranée. Il en résulte la topographie complète du gouffre reproduite à la Figure 2-11.

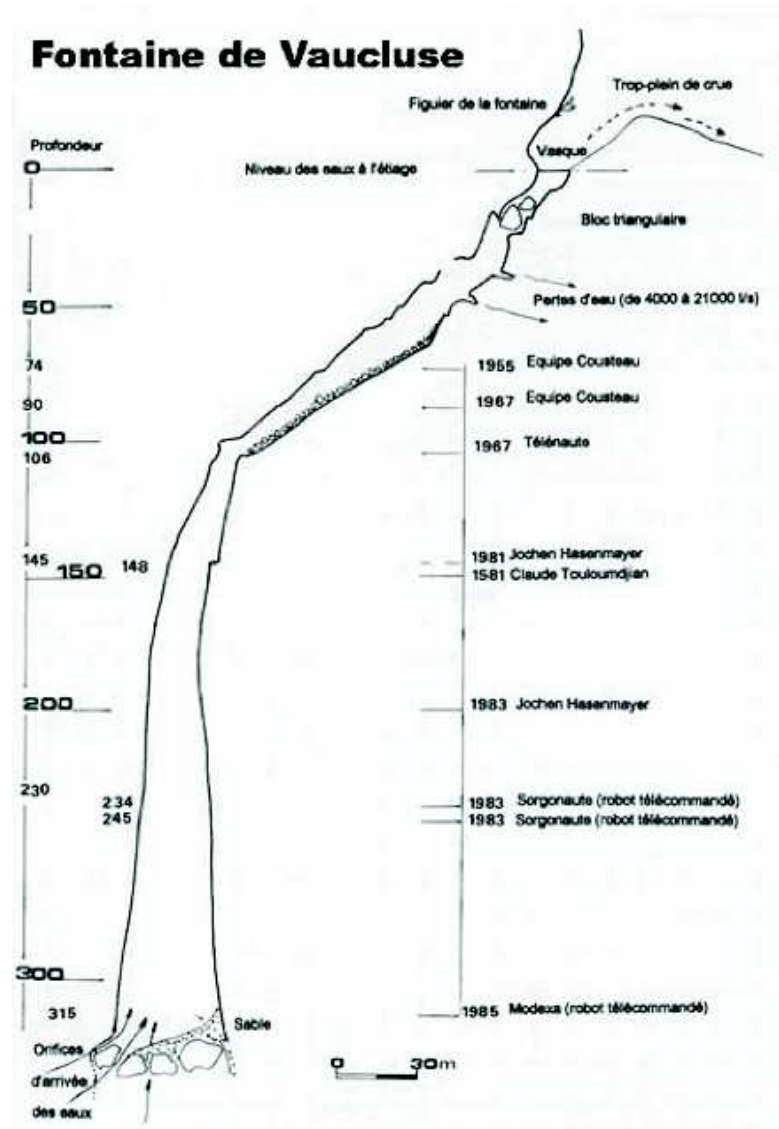


Figure 2-11 : Topographie de Fontaine de Vaucluse [30]

En 1986, un Sorgonaute III fut construit afin de récupérer le Sorgonaute II. L'affaire tourne au cauchemar et malgré tous les efforts l'engin est perdu à son tour, laissant derrière lui 150 mètres de câble coincés dans le gouffre. Sorgonaute IV a été mis à l'eau en août 1988 pour également tenter une opération de sauvetage afin de récupérer ses prédécesseurs. La mission fut également un échec [16].

Le Spélénaute fait son apparition en 1989, il est né d'un partenariat entre la Comex et la société C.M.C. Mahieu. Cet engin est le prototype de la série Achille construit par la Comex.

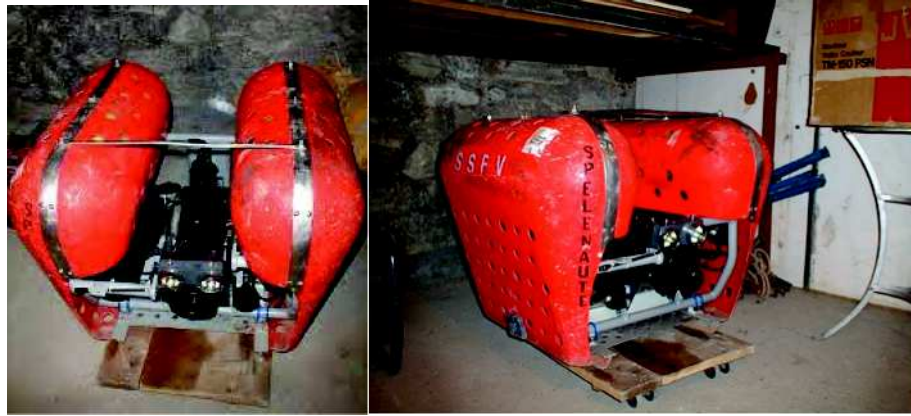


Figure 2-12: Spélénaute (1989) [33]

Le rôle du Spélénaute (Figure 2-12) va plus loin que la simple exploration réalisée par ces prédécesseurs. Ce ROV va permettre de dresser une topographie plus complète, de prendre des images, de mesurer le débit d'eau, de faire une étude du fond ainsi que de faire des prélèvements d'échantillons d'eau, sable, etc.

Lors d'une plongée en 1996 dont le but était de mesurer la section de la galerie, le Spélénaute est resté accroché dans le câble du Sorgonaute III qui lui est resté coincé en 1987 par -233 mètres. Un engin de la Comex équipé d'une disqueteuse est alors intervenu pour couper les différents câbles, cet engin a également été bloqué à une profondeur de 160 mètres le lendemain lors du second essai qui devait permettre la libération du premier engin perdu. Les deux engins ont été récupérés par une équipe de plongeurs [8] après l'échec du sauvetage par la Comex.

Pour réaliser ces opérations complexes, les vecteurs doivent emporter un grand nombre de capteurs tels un sonar profilométrique, un sonar DVL, une caméra acoustique, un capteur CTD, une caméra HD, etc... représentant un encombrement conséquent (sans omettre le poids et la consommation). De plus, un contrôle avancé est nécessaire. Parmi les limitations de ces solutions d'explorations, nous retiendrons ici deux caractéristiques problématiques : leur volume et la présence du cordon ombilical.

Ces explorations montrent en effet que le cordon ombilical est un point critique et probablement le point bloquant. Il est *a priori* impensable de se passer du cordon ombilical dans la phase de découverte du milieu, phase durant laquelle l'homme du métier doit pouvoir piloter le vecteur, en fonction de sa propre expertise. Mais ce cordon pourrait être « décroché » dans la phase de retour si le vecteur est capable d'exploiter la connaissance acquise dans la phase aller. L'idée sous-jacente est donc de passer d'un vecteur ROV à un vecteur AUV. Cette application est un véritable défi scientifique et technologique retenu comme objectif à long terme, mais qui influence grandement notre réflexion sur la polyvalence.

D'autres explorations de gouffres ont été réalisées à l'international.

2.1.4.2 Diverses explorations karstiques robotisées

L'exploration karstique avec des robots sous-marins n'est pas réalisée uniquement en France, on voit également des missions se monter dans les années 2000 en Italie [34], dans le gouffre Pozzo del Merro (Figure 2-13).

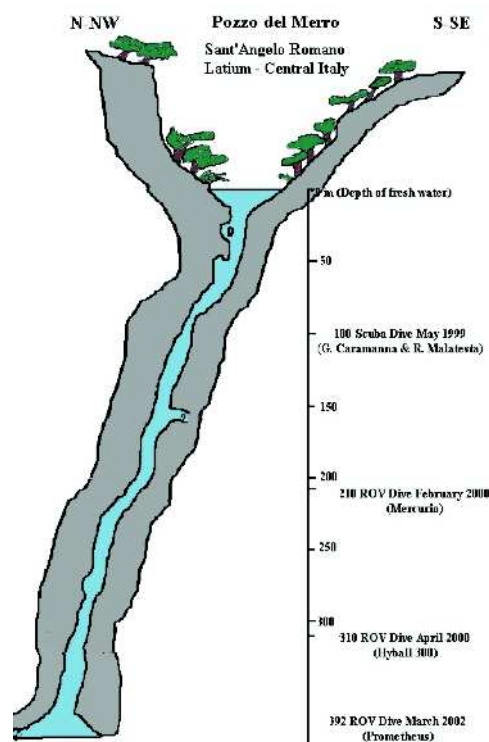


Figure 2-13 : Topographie du gouffre Pozzo del Merro[34]

La première fut en février 2000 avec le Robot « Mercury » avec une descente à -210 m et une seconde mission en Avril 2000 avec le « Hyball 3000 » (Figure 2-14). Le Hyball s'approcha du record du monde détenu par le Spélénaute depuis 1985 (-315 m) en descendant à -310 m sans problème.



Figure 2-14 : A gauche le Mercury et à droite le Hyball 300 (2000) [34]

Il faut attendre 2002 pour qu'un ROV établisse un nouveau record du monde, ce record est toujours détenu par le Prometheus (Figure 2-15) avec une profondeur de -392 m [34].

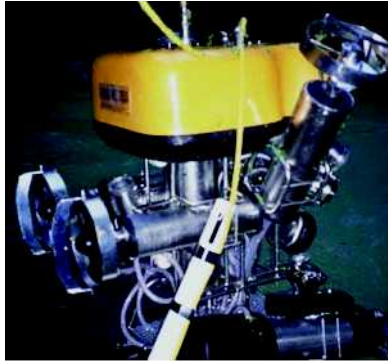


Figure 2-15 : Prometheus (2002) [34]

En 2013, une équipe d'hydrologues a utilisé le ROV Seaeye du constructeur Saab pour l'exploration du gouffre Crveno Jazero en Croatie [9]. Le Seaeye est un ROV haut de gamme permettant un relevé topologique de qualité grâce à son sonar profilométrique (Figure 2-16).



Figure 2-16 : Seaeye de Saab (2013) [35]

Depuis le début de l'utilisation des robots sous-marins en spéléologie leur taille est habituellement réduite, ceci pour faciliter leur évolution dans l'environnement. Mais ce n'est pas toujours le cas. En mars 2007 la NASA sort un engin nommé DepthX (Deep Phreatic Thermal Explorer) [36], équipé de nombreux capteurs pour l'analyse de l'eau (Figure 2-17). Ce robot de forme sphérique qui pèse 1.5 tonnes pour un diamètre d'environ 2 m a été utilisé en 2007 dans le cénote de Zacaton [37]. Les cénotes sont des puits verticaux naturels, produits par un phénomène de dissolution et d'effondrement des terrains calcaires situés au-dessus d'un réseau souterrain de grottes et rivières. Ce type de puits peut faire des centaines de mètres de profondeur. Le fond du cénote Zacaton est situé à -319 m de la surface [37]. L'exploration d'un cénote est plus facile du fait qu'il s'agit d'un large cylindre vertical contrairement aux réseaux karstiques qui peuvent contenir des tronçons horizontaux et des siphons.

Le DepthX contrairement aux autres robots n'est pas un ROV mais un AUV.



Figure 2-17 : DepthX [36]

Ce robot est une plateforme scientifique avec une logistique lourde. Il embarque une multitude de capteurs et d'unités de traitement : 54 sonars, 36 unités de traitement, centrales inertielles, loch doppler, caméra, microscope, etc.

Le robot est également équipé d'un bras articulé lui permettant de faire des prélèvements de roche, sable, eau, etc. Les 54 sonars lui permettent de refaire une reconstruction 3D du cénote (Figure 2-18) [37].

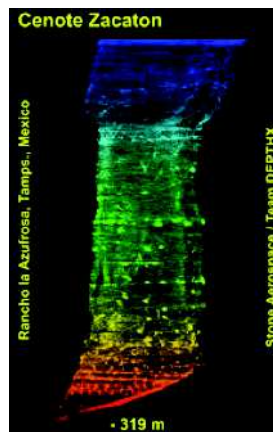


Figure 2-18 : Reconstruction 3D [37]

Ces capacités sont bien évidemment supportées par la grande taille de l'engin et l'absence du câble facilite l'évolution de l'engin en limitant le risque de perte de celui-ci. Mais ce genre de vecteur n'est pas utilisable dans la plupart des réseaux karstiques. L'exploration des karsts requiert des engins de petite taille.

Cette présentation non exhaustive de plusieurs applications du domaine de l'exploration faible fond ou confinée, montre l'importance des capteurs à mobilité contrôlée et de la polyvalence nécessaire pour répondre au plus grand nombre d'applications. Déclinons ces propos sur notre vecteur.

2.2 Un vecteur robotique polyvalent

Comme nous l'avons évoqué en illustrant diverses applications, la polyvalence du robot impose que l'on puisse l'adapter selon la charge utile (capteurs à embarquer), la puissance

d'actionnement nécessaire (notamment selon le milieu dans lequel il porte les capteurs) et l'intelligence embarquée requise (capacité opérationnelle). Ces considérations conduisent à concevoir de manière cohérente les différentes architectures du robot, à savoir : l'architecture mécanique, l'architecture matérielle (au sens des cartes électroniques) et les architectures logicielles (informatique et automatique). L'évolutivité du vecteur est en effet à projeter sur ces quatre « architectures ».

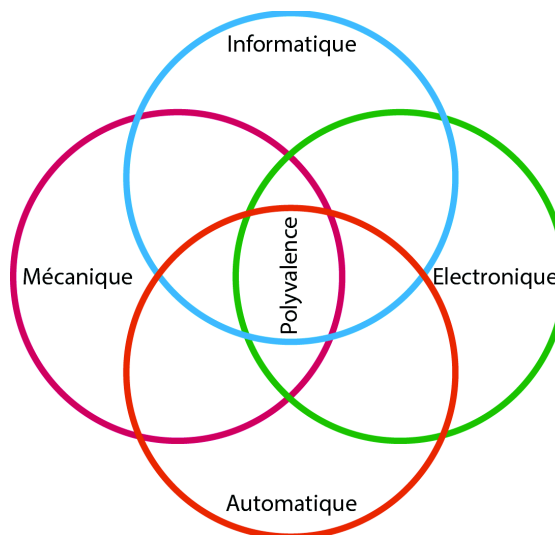


Figure 2-19 : Projection de la polyvalence

2.2.1 Le vecteur

Qu'il s'agisse du niveau national ou international, l'offre technologique se cantonne à quelques véhicules légers sur une grande majorité de gros robots sous-marins¹⁹ comme nous l'avons mentionné. Au niveau européen seuls trois industriels ont dans leur gamme des ROV légers (Ageotec²⁰-Italie, Rovtec²¹-Angleterre, MicroRovServices²²-Angleterre) et proposent les services d'opération associés. D'autres acteurs se positionnent sur la fabrication de robots sous-marins légers de type ROV (Ciscrea et SubSea-Tech – France, Saab – Suède, SeaBotix²³ et Bluefin Robotics²⁴ – US, pour citer les principaux) sans proposer les services associés. Il est à noter que la gamme des 'petits' ROV comprend pour la plus grande partie des systèmes d'un poids supérieur à 50 kg, ce qui occasionne une logistique restant conséquente.

Dans le cadre de nos travaux, nous visons la conception d'un vecteur dont l'intelligence embarquée, la charge utile et les capacités propulsives puissent évoluer en fonction des opérations visées, faisant ainsi abstraction des « problématiques robotiques » liées au système et facilitant son exploitation. De plus certaines des applications que nous envisageons (dont l'exploration karstique) requièrent un traitement avancé des données mesurées pour reconstruire des modèles d'environnement exploitables par la commande du système ; par conséquent, il est également nécessaire de pouvoir ajouter de la puissance de calcul selon le besoin.

¹⁹ Panorama partiel de l'offre : www.rov.org/industry_all.cfm

²⁰ www.ageotec.com

²¹ www.rovtechsystems.co.uk

²² www.microrovservices.com

²³ www.seabotix.com

²⁴ www.bluefinrobotics.com

2.2.2 Morphologie du vecteur

L'élément de base est le robot Jack développé par l'entreprise Ciscrea. Le Jack est l'un des seuls ROV de sa catégorie à appuyer son architecture mécanique autour de deux flasques horizontales ; la plupart des autres ROVs utilisent des flasques verticales. Le fait d'utiliser des flasques en position horizontale ouvre de plus grandes perspectives d'évolutivité. Cette architecture permet l'emport de flottabilité sur la partie haute de l'engin et de la masse sur la partie basse, elle permet également de faire de la reprise d'effort dans le plan horizontal et vertical ainsi que de protéger le robot lors de son utilisation, puisque ses formes arrondies permettent de réduire les risques de l'accrocher (Figure 2-20). Entre les deux flasques, on retrouve un tube en aluminium rendu étanche grâce à deux tapes en plastique (PVC). La tape avant est équipée d'une bulle en polycarbonate permettant l'orientation de la caméra interne, sur la tape arrière se trouve l'ensemble des passages de coques. Dans cette version, les tapes et le tube central garantissent une étanchéité jusqu'à 100 m de profondeur. L'architecture mécanique de base permet l'interconnexion de caissons via de la connectique étanche standardisée, positionnée également sur la tape arrière.

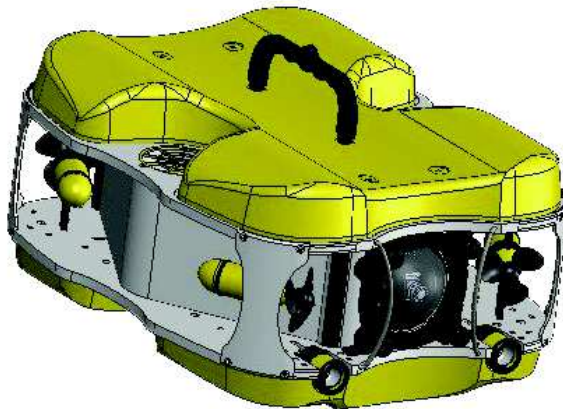


Figure 2-20: Version de base du robot Jack

Dans cette version de base, le robot embarque un étage d'actionnement constitué de 6 moteurs, dont l'étude sera traitée dans le chapitre 3. L'engin est équipé d'un ensemble de capteurs permettant la mesure de température interne et externe, la détection d'entrée d'eau, la mesure du cap (compas) et la mesure de pression afin de connaître la profondeur de l'engin. Il est également pourvu de moyens d'éclairage pilotables, d'une caméra, et enfin d'une simple carte de contrôle (bas-niveau) permettant de supporter les processus d'acquisition des données capteurs, de contrôle des actionneurs ainsi qu'un lien de télé-opération.

Ces aspects relatifs aux architectures matérielle et logicielle de contrôle seront explicités par la suite, notamment car elles ont été intégralement modifiées lors de nos travaux. En effet, nous sommes partis de cette version de base pour concevoir un vecteur polyvalent, capable d'embarquer une variété de capteurs « mission » selon l'application visée, dont :

- centrale inertielle (mesure de l'attitude du robot),
- sonar profilométrique (image sonar du profil de coupe d'un canal, d'un aquifère),
- sonar sédimentaire (image des contrastes de densités présentes dans le sédiment),
- sondeur multifaisceaux (bathymétrie),

- magnétomètre (détection d'artefacts magnétiques du milieu),
- loch doppler (mesure des courants principaux ou de fuite),
- capteur CTD (propriétés physico-chimiques du milieu),
- caméras (par ex. caméra acoustique 2D pour vision en eau turbide), etc.

Pour ce faire, la version de base peut être simplement modifiée par l'ajout d'un châssis, appelé « skid » (berceau d'emport de capteurs), permettant d'embarquer des capteurs encombrants (Figure 2-21).

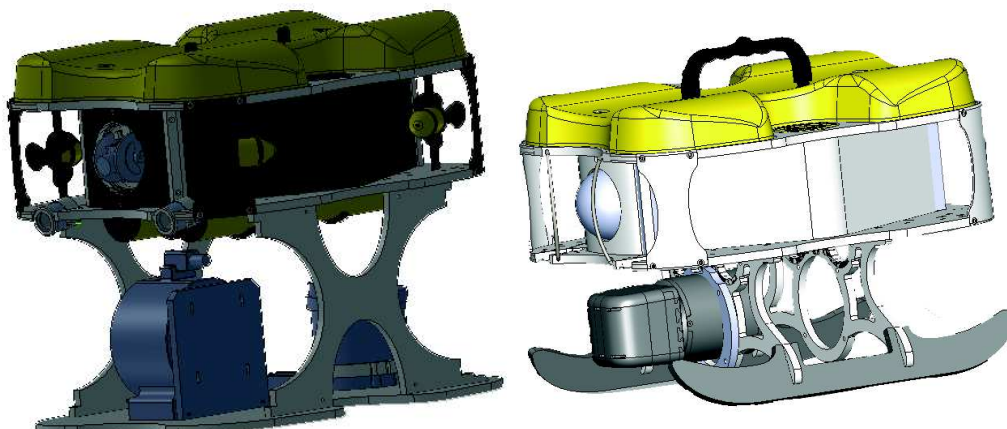


Figure 2-21: Le robot jack équipé avec deux versions de skid.

Les capteurs acoustiques mentionnés sont tous de volume, poids et consommation très conséquents (allant de 2 à 5 kg, de 0.002 à 0.016 m^3 , pour une consommation atteignant jusqu'à 2.5 A (30W)).

Marque	Modèle	volume	Poids en air	Poids en eau	Consommation
Imagenex	Delta-T	0.002 m^3	2.5 kg	0.7 kg	12W
Tritech	Super SeaKing	0.010 m^3	3.5 kg	1.7 kg	24W
Blueview	M900-90	0.006 m^3	2.3 kg	0.5 kg	13 W
Rowe	SeaPilot	0.016 m^3	5.0 kg	-	11 W
EvoLogics	S2CR 48/78	0.006 m^3	2.1 kg	1.4 kg	30 W

Tableau 2.1 : Caractéristiques techniques des capteurs du LIRMM pouvant être embarqués sur le robot.

Un des inconvénients de ce skid est son impact sur la dynamique du vecteur car il induit des couplages non désirés de la dynamique des degrés de liberté (DDL) du système. Un positionnement fin requiert d'accroître les capacités d'actionnement du système (Figure 2-22). De fait le système devient alors fortement sur-actionné, et la gestion dynamique de cette redondance doit permettre un gain conséquent en précision, réactivité et robustesse (traité dans le Chapitre 3).

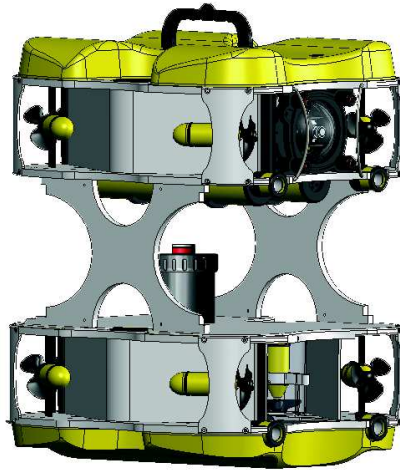


Figure 2-22: Le robot Jack équipé d'un skid et de son extension d'actionnement.

Dès lors que le vecteur doit accomplir des missions de façon autonome, ou partiellement autonome, il est nécessaire de le doter de ressources de calcul supplémentaires. Un caisson « contrôle haut-niveau » doit être ajouté au vecteur (Figure 2-23). La connectique devient un point important pour préserver l'évolutivité du vecteur. Ce point sera donc ré-évoqué plus loin.

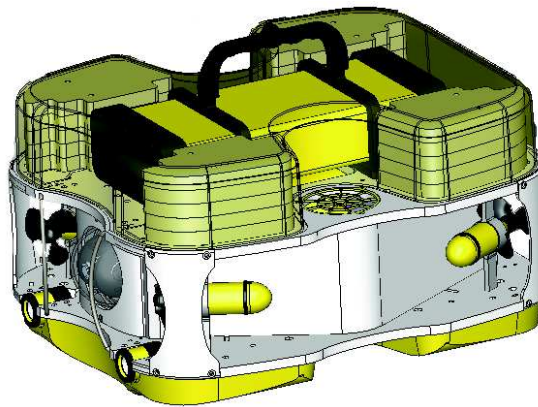
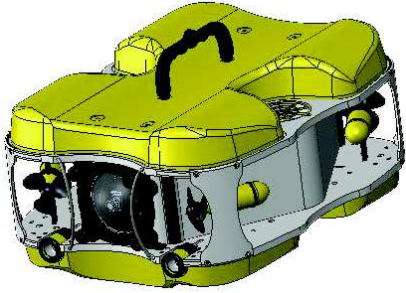
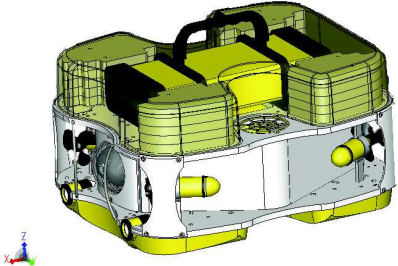
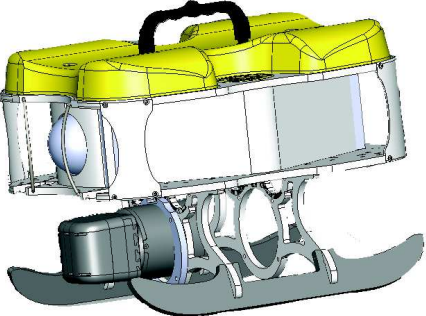
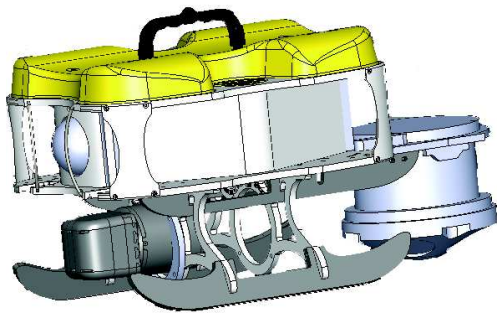


Figure 2-23: Le robot Jack équipé d'une électronique supplémentaire vouée à supporter un contrôle plus évolué

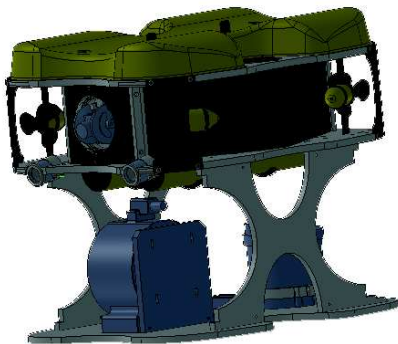
Nous pouvons constater que la morphologie du vecteur peut, selon cette conception, se décliner en une **gamme de produits**. Chaque produit de cette gamme permet d'adapter le vecteur aux besoins de l'application. Les versions conceptuelles de la plateforme que nous proposons sont présentées dans le Tableau 2.2. L'objectif n'est pas de concevoir une version générique qui peut répondre à l'ensemble des besoins, mais bien une version adaptable aux besoins. C'est par cette adaptation que nous visons la polyvalence, sans prétendre bien sûr pouvoir répondre à toutes les applications imaginables. Les besoins de polyvalence dans la morphologie de l'engin amènent néanmoins à des problèmes, des contraintes ou des limitations de la solution.

<i>Jack de base</i>	
	<p>La version de base, avec ses capteurs intrinsèques et son contrôleur bas niveau, limite le vecteur à un ROV. Téléopérable, son intelligence embarquée est limitée, par exemple à l'asservissement d'une orientation (ψ) et d'une profondeur (z). Le jack de base peut être utilisé dans des applications où l'opérateur assure l'essentiel du pilotage et l'intégralité de la charge décisionnelle. Ce qui induit un pilotage difficile qui requiert de l'expertise. L'opérateur peut être cependant être aidé par des traitements déportés, mais qui nécessitent un lien ombilical.</p>
<i>Jack de base + Contrôle haut-niveau</i>	
	<p>Cette version est une extension basée sur l'ajout d'un caisson dédié à un contrôle haut-niveau (CHN), mais elle reste similaire à la version de base en termes de capacité de charge utile. Les capacités de calcul supplémentaires permettent de centraliser sur le robot des commandes avancées qui, dans la version de base, ne pouvaient être supportées que par le poste opérateur. Dans cette version le vecteur peut être utilisé en mode AUV, avec de faibles capacités d'emport.</p>
<i>Jack de base + Skid luge + Caméra acoustique</i>	
	<p>Dans cette version, le robot est équipé d'un « skid luge » afin d'embarquer une caméra acoustique (comme par exemple la Blueview de Teledyne, dans notre cas). Cette version permet d'embarquer une charge utile mais sans CHN. Les possibilités de contrôle restent celles de la version de base ; les données capteur sont collectées mais pas exploitées en ligne, à l'exception de traitement « simple » comme la détection d'une paroi dans l'axe frontal du robot (obstacle). Dans le cas de l'emport du CHN, les données capteur peuvent être exploitées pour un contrôle plus évolué, dont par exemple un suivi de paroi.</p>
<i>Jack de base + Skid luge + Caméra acoustique + DVL</i>	



Cette version embarque en plus un capteur de vitesse (DVL à effet doppler « SeaPilot» de Rowe, dans notre cas), permettant d'atteindre un meilleur référencement relatif par rapport au milieu (vitesse et élévation du robot par rapport au fond). Une fois de plus en cas d'emport d'un CHN un contrôle avancé peut être embarqué, utile par exemple dans le suivi (transect) de récifs de corail en présence de courant. L'inconvénient d'un tel emport de capteur est évident, notamment en termes d'impact sur la dynamique de l'engin.

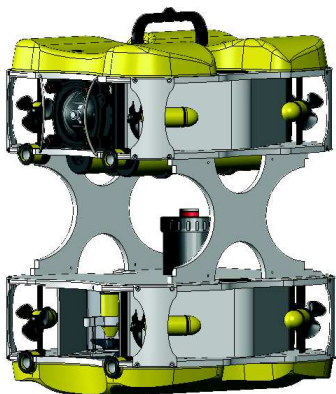
Jack de base + Skid panier + Sonar bathymétrique



Cette version du jack est équipée d'une autre version du skid, plus haute, pour permettre l'emport d'un capteur mission de volume plus conséquent comme un sonar bathymétrique par exemple. Ce sonar bathymétrique est un sonar multifaisceaux, qui permet une visualisation en 2D (le sonar Delta-T de Imagenex, dans notre cas). La même remarque s'applique concernant l'emport ou non d'un CHN ; l'exploitation en ligne des données peut être utile pour l'inspection de voie navigable, par exemple. De par son encombrement, la dynamique de l'engin est également affectée.

Ces cas précédents montrent l'importance d'accroître l'actionnement du vecteur.

Jack de base + Skid sandwich + Extension d'actionnement + DVL



Comme nous avons pu le constater à travers les versions précédentes, l'emport de capteurs acoustiques de taille et de poids conséquents est difficilement réaliste au regard de l'impact sur la dynamique et le comportement du vecteur. De fait, nous intégrons dans notre conception de la gamme une version avec extension d'actionnement, pour compenser également le sous-actionnement de la plateforme. Cette extension d'actionnement permet d'ajouter de la puissance motrice à l'engin et de rééquilibrer le vecteur lors de ces déplacements (6 degrés de liberté actionnables, au lieu des 5 dans la version de base). Cette extension embarque également sa propre énergie, ce qui permet d'adapter aussi les capacités énergétiques. Cette version est plus orientée pour une utilisation en

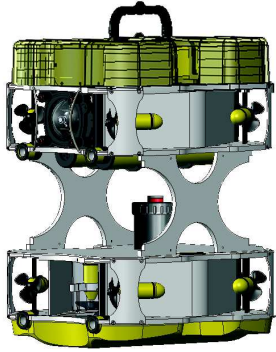
	<p>exploration et en cartographie de milieu karstique, du fait de sa redondance d'actionnement, nous gagnons en manoeuvrabilité et robustesse, permettant d'orienter les capteurs par rapport au terrain.</p>
<p><i>Jack de base + Skid sandwich + Extension d'actionnement + DVL + CHN</i></p>	
	<p>Cette version est la version la plus évoluée de la gamme, c'est la configuration la plus optimale pour l'exploration et la cartographie des réseaux karstique. Cette configuration en plus d'avoir une capacité d'emport de capteurs importante grâce à son skid, est également très manoeuvrable du fait de son extension d'actionnement. Elle embarque aussi le CHN afin de réaliser des lois de contrôle plus évoluées dans le cas d'une autonomie partielle, voire complète (câble ombilical non présent), du robot.</p>

Tableau 2.2 : Gamme de produit

Cette évolution « mécanique » doit être accompagnée d'une évolution cohérente de l'architecture de contrôle embarquée, à savoir les architectures matérielle et logicielle. L'architecture matérielle est entendue au sens de l'architecture issue de la composition des cartes électroniques embarquées. Ces « cartes » sont distinguées au regard des fonctions qu'elles supportent. Rappelons que la préoccupation est relative à l'évolutivité, et donc à la possibilité d'ajouter des cartes selon la version du robot, à savoir une version plus ou moins évoluée en termes d'autonomie opérationnelle. Il s'agit de la capacité à faire ou à conduire seul des actions évoluées comme, par exemple, assurer le contrôle en profondeur (faire) et enchaîner un ensemble d'actions selon la mission définie (conduire).

2.2.3 Architecture matérielle (électronique)

La proposition illustrée sur la Figure 2-24 est donc basée sur une carte « contrôle bas-niveau » (CBN) au sein de laquelle peuvent être déployées des fonctions multiples permettant à l'opérateur de piloter le vecteur en donnant simplement des consignes (par ex. la profondeur à laquelle le vecteur doit être stabilisé). Le CBN permet également un contrôle direct des actionneurs, dans un mode que l'on pourrait qualifier de « transparent », que l'on évoquera plus tard. La carte CBN doit également être connectable au caisson d'extension de l'actionnement (cf. les différentes versions évoquées), avec son ensemble de moteurs supplémentaires.

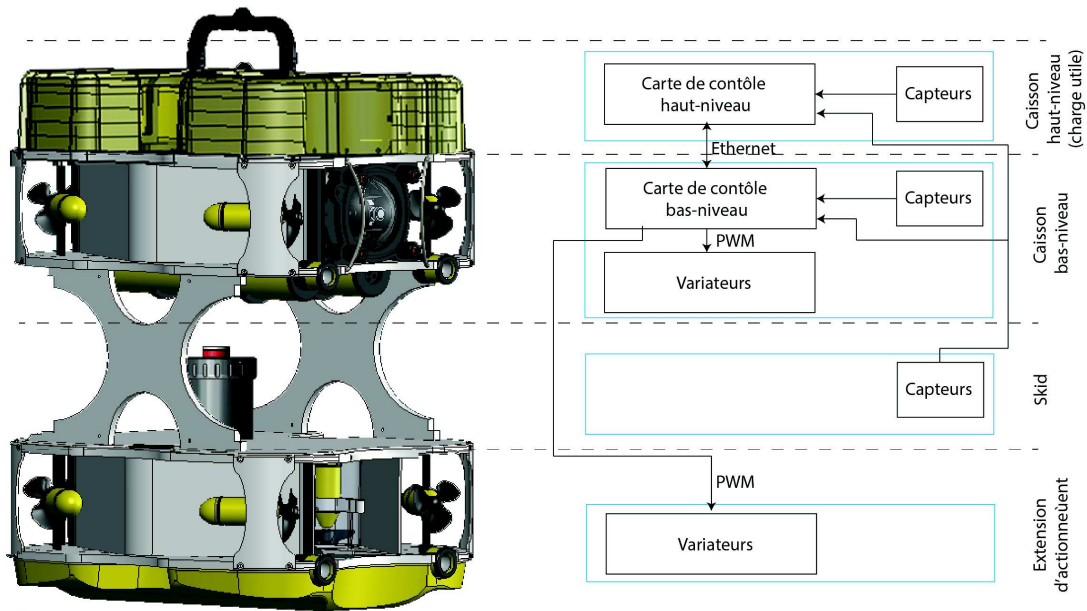


Figure 2-24 : Description symbolique de l'architecture matérielle du robot

La réalisation d'un vecteur robotique plus évolué, au sens d'un vecteur doté d'une capacité décisionnelle et opérationnelle plus importante, passe par l'ajout d'une carte de « contrôle haut-niveau » (CHN) en charge des fonctions correspondantes comme la gestion de mission, la gestion de mode de fonctionnement, des fonctions de contrôle plus évoluées (dont les algorithmes requièrent une puissance de calcul supérieure à celle offerte par la carte de CBN), voire même des traitements d'images vidéos ou sonar. Le lien entre les deux cartes de contrôle, à savoir CBN et CHN, est important ; d'une part il ne doit pas introduire de latence conséquente, inhibant sinon la possibilité d'un contrôle du robot par la carte de haut-niveau, et d'autre part il doit permettre différentes connexions selon la version.

En effet, cette architecture matérielle est « connectée » ou non, à l'opérateur selon le type de vecteur que l'on considère :

- Soit un vecteur de type ROV, dans ce cas il y a un « cordon ombilical » entre le poste opérateur et le robot. Deux types de configurations sont dès lors possibles selon la version du vecteur : un lien de télé-opération connecté au CBN (Figure 2-25), permettant un pilotage direct de l'étage d'actionnement, ou un lien de télé-opération connecté au CHN (Figure 2-26) offrant ainsi à l'opérateur des services évolués et une supervision fine.

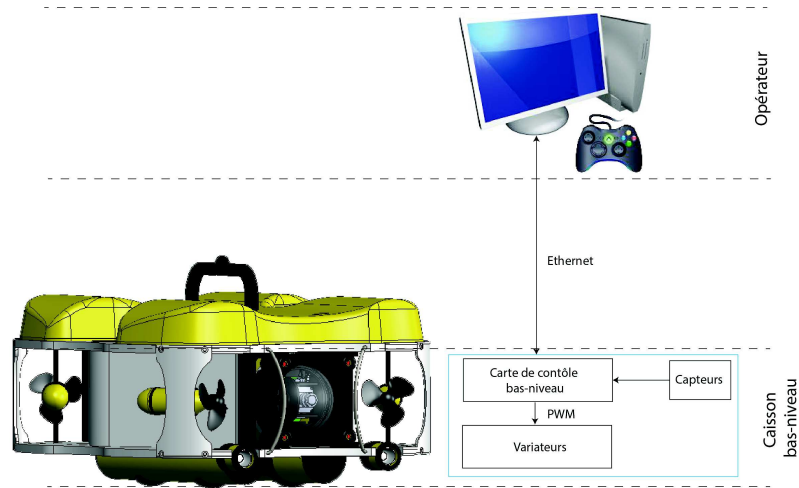


Figure 2-25 : Lien de télé-opération "bas-niveau"

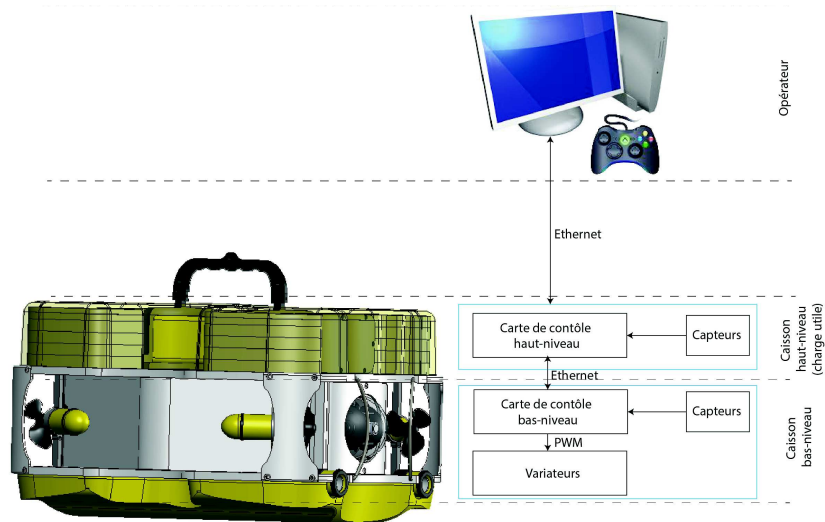


Figure 2-26 : Lien de télé-opération "haut-niveau"

- Soit un vecteur de type AUV et dans ce cas il n'y a pas de cordon de télé-opération (Figure 2-27). Le système est autonome et seule une supervision réduite est possible, via une communication acoustique faible débit et au mieux intermittente. Un des enjeux se situe dans l'analyse en ligne des données capteurs pour fournir des modèles d'environnement exploitables par l'étage décisionnel du robot. Ce point souligne l'importance de l'architecture du système qui doit être en mesure de recueillir la connaissance « métier » (i.e. cartes *a priori*) de l'utilisateur et l'exploiter à des fins de « navigation éclairée ».

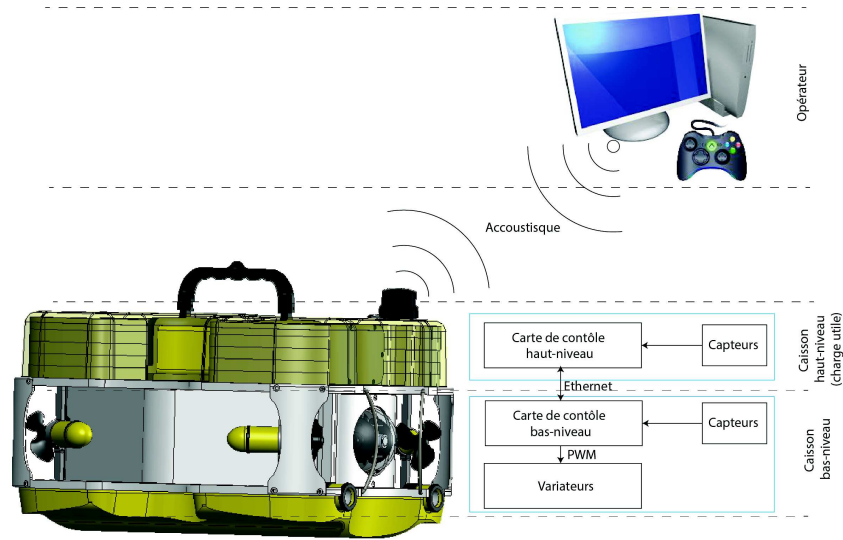


Figure 2-27 : Sans lien de télé-opération (autonome)

Le principe est d'exploiter autant que possible des cartes électroniques du commerce afin d'éviter de développer des solutions « propriétaires » parfois peu évolutives, et nécessitant souvent un investissement conséquent (design, prototypage, validation). De plus, l'évolution technologique est conséquente et donc une solution propriétaire représenterait un risque de rapide d'obsolescence.

En considérant les différents points évoqués plus haut, notre choix s'est porté sur une BeagleBone Black. Cette carte du commerce est équipée d'un processeur de type ARM et d'une capacité stockage nécessaire pour l'utilisation du patch temps réel Xenomai et du middleware ContrACT développé au LIRMM [38], [39] (Figure 2-28).

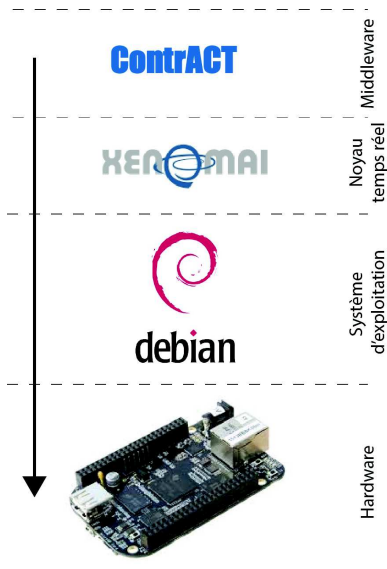


Figure 2-28 : Architecture logicielle

Cette architecture hardware (Figure 2-29) contient un capteur de pression pour la mesure de la position du robot par rapport à la surface (z), une centrale inertielle permettant de connaître les orientations $\{\phi, \theta, \psi\}$ (radian) ou de son quaternion \mathbf{Q} , une estimation grossière des accélérations $\{u, v, w\}$ (m/s) suivant X, Y et Z ainsi que les vitesses de rotations $\{p, q, r\}$ (rad/s) suivant u, v et w du robot.

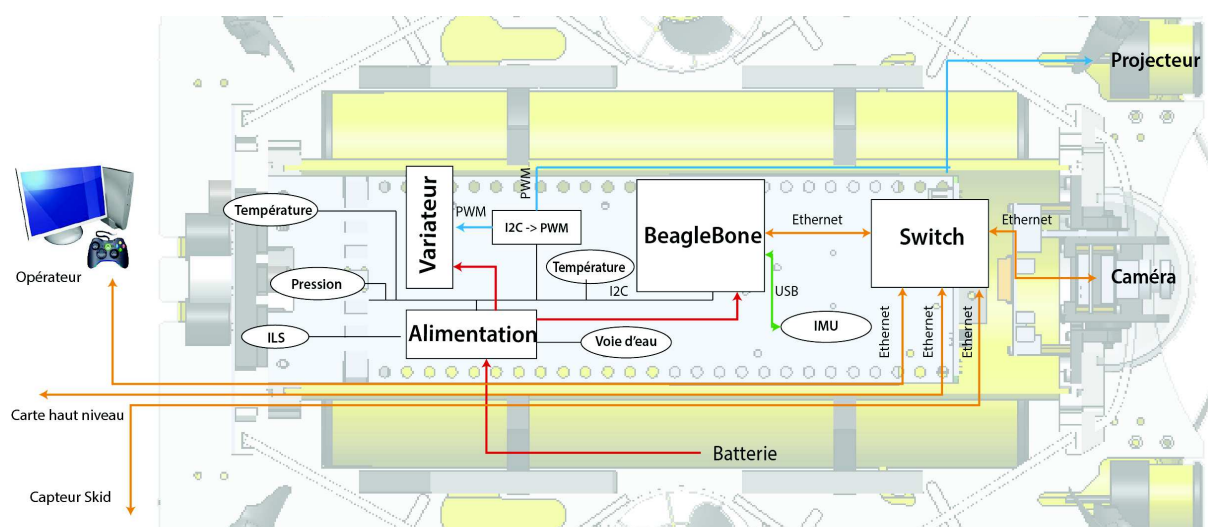
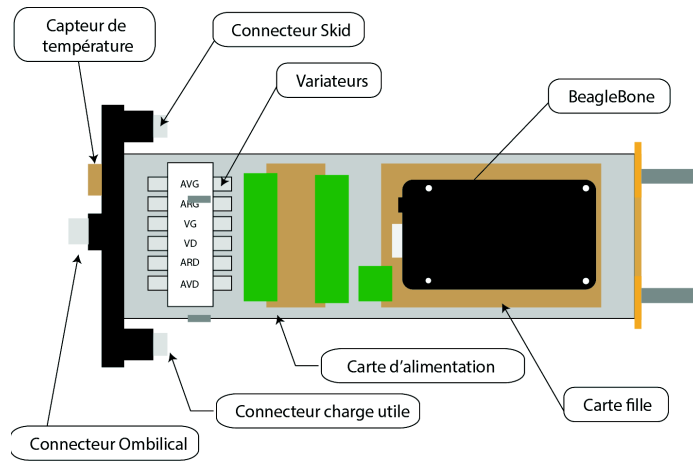


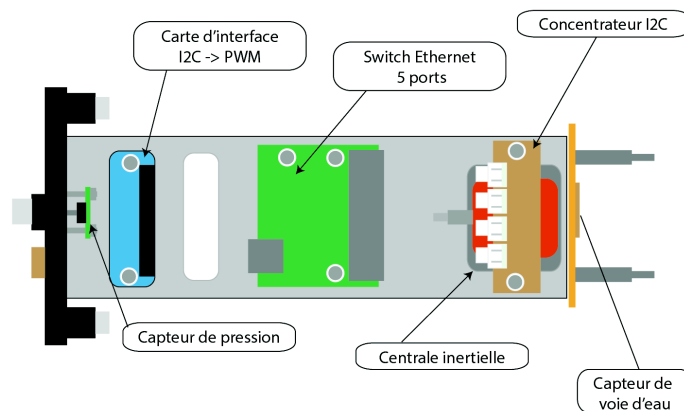
Figure 2-29 : Architecture hardware du Jack

Cette électronique est alimentée par deux batteries de 9 A/h. Chacune d'elles est équipée d'un capteur de courant et de tension permettant de mesurer la consommation électrique instantanée du robot. Pour la sécurité de la plateforme, celle-ci est équipée d'un capteur d'entrée de voie d'eau pour la détection d'eau ou de condensation dans le caisson étanche. De même, plusieurs capteurs de température sont placés à divers emplacements de cette même enceinte, à l'intérieur et l'extérieur (température de l'eau). L'un de ces capteurs internes est positionné au plus près des variateurs. Les variateurs permettent le contrôle des moteurs de type brushless (moteur électrique sans charbon). Le contrôleur permet de faire une commutation de courant sur les 3 phases du moteur créant ainsi un champ magnétique tournant sur lequel les pôles de la cloche du rotor viennent s'aligner.

La figure suivante décrit l'emplacement des différents composants de l'électronique embarquée.



(a) Vue de dessus



(b) Vue de dessous

Figure 2-30 : Implantation de l'électronique embarquée

En termes de réseau embarqué, le choix s'est porté sur une solution Ethernet commuté dans la mesure où : d'une part cette technologie facilite l'interconnexion des capteurs « métier » (charge utile) avec les cartes de contrôle selon des performances adéquates (en termes de débit) et d'autre part de nombreux capteurs acoustiques commercialisés sont communicants sur Ethernet. Certes le réseau Ethernet n'est pas déterministe, mais il a été privilégié par rapport aux multiples standards industriels existants au regard des critères d'évolutivité, du besoin de bande passante et de son adéquation aux communications IP.

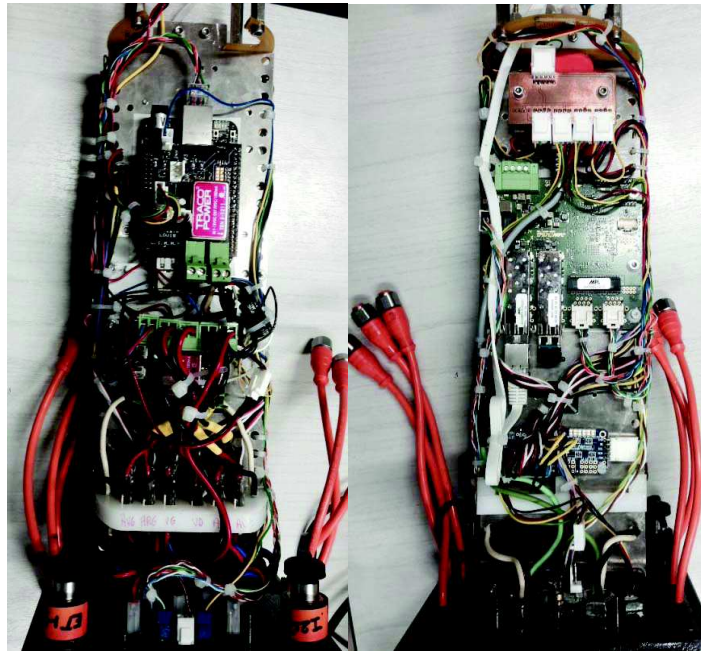


Figure 2-31 : Photos de l'électronique bas niveau

Le robot est également équipé d'une caméra communicant sur Ethernet (Figure 2-33), afin d'être homogène avec le réseau Ethernet commuté embarqué sur lequel sont connectés les différents capteurs « métier » (Figure 2-32). L'annexe II décrit la configuration réseau de l'architecture.

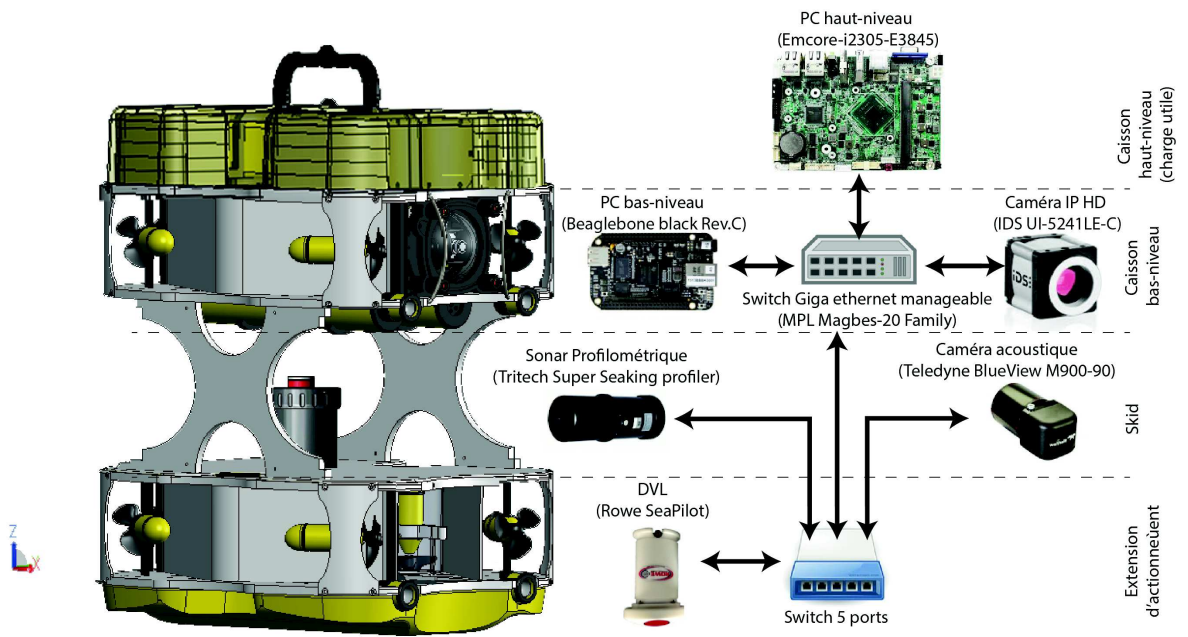


Figure 2-32 : Interconnexions des équipements communicants en Ethernet.

Son flux de données numériques (caméra full HD) peut être exploité dans le cadre d'un contrôle basé sur un asservissement visuel (suivi de canalisation, évitement d'obstacles, etc.)

ou directement par l'opérateur « métier » capable d'interpréter les images observées (hydrogéologue, archéologue, etc.).

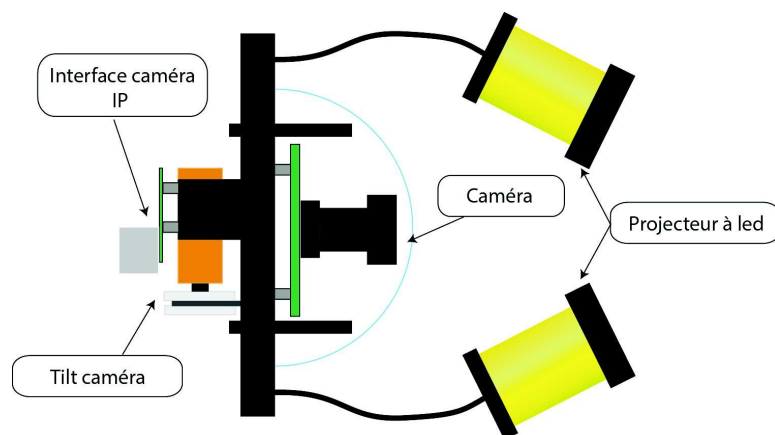


Figure 2-33: Tape avant du robot

2.2.4 Architecture de commande

Au regard des différentes applications possibles, l'architecture de commande doit comprendre un ensemble de lois de commande répondant aux différentes fonctionnalités requises, et ce en se basant potentiellement sur des charges utiles différentes. Illustrons quelques cas en nous appuyant sur les exemples d'applications métiers que nous avons précédemment mentionnés :

- Dans sa version de base, le vecteur embarque des capteurs en mesure d'informer l'opérateur dans sa mission de guidage, mais c'est l'opérateur qui assure le contrôle du vecteur. Ce contrôle se résume souvent à la transmission de consignes moteurs. Néanmoins, il n'est pas évident pour un opérateur de raisonner ainsi, ni même de sélectionner les moteurs nécessaires au mouvement désiré et de maîtriser les efforts induits. Idéalement, le contrôle des moteurs devrait être transparent, afin de reproduire au mieux les intentions de l'utilisateur. Ainsi, l'ensemble des moteurs devra agir de manière conforme aux attentes de l'utilisateur, en fonction de la consigne d'immersion de l'opérateur via l'IHM, mais ce point là ne sera pas traité.
- Les versions plus « évoluées » du système embarquent un actionnement plus puissant, et de par la réutilisabilité recherchée, nécessairement redondant. Cette redondance peut être exploitée avec de multiples objectifs et permet d'attribuer à l'étage d'actionnement des propriétés de robustesse et de manœuvrabilité précieuses. De plus, dans une telle situation, un pilotage direct des moteurs par l'utilisateur est impossible. La réification de l'étage d'actionnement et de ses propriétés, permettant d'abstraire la question de la génération du mouvement désiré – issu de l'opérateur ou des multiples lois de commande nécessaires aux différentes versions du vecteur – est un point clef de la polyvalence, traité au chapitre 3.
- La diversité des applications qui attendent notre système engendre une multitude d'algorithmes de traitement, de navigation, de guidage, de contrôle, de supervision... dont l'exécution doit être rigoureusement orchestrée et, bien sûr, adaptée aux différentes

architectures matérielles (mécanique, électronique et des calculateurs) requises par l'application. Le développement de ces algorithmes n'est pas l'objet de cette thèse. Nous avons cependant profité des travaux de A. Lasbouygues [40] sur la commande (formalisme atomes), et de ceux de S. Louis sur la simulation (formalisme quaternions), tous deux doctorants du laboratoire.

Pour ce faire nous introduirons une solution basée sur le contrôle des degrés de liberté du vecteur (DDL) s'appuyant sur une entité appelée « répartiteur » en charge de la correspondance entre les DDL et les moteurs. Ainsi, grâce à cette approche l'opérateur, en situation de téléopération directe, pilotera directement les DDL du système, mais devra guider le vecteur selon l'analyse que ses compétences propres lui permettent de faire des données qui lui sont fournies. Ici l'intelligence du système réside dans celle de l'opérateur et s'appuie sur la pertinence des informations fournies. Un exemple classique est une téléopération directe avec retour vidéo. Cependant, la qualité d'*immersion* reste limitée et il est difficile d'appréhender les mouvements à induire sur le système, ne serait-ce par exemple que pour stabiliser l'image dans un environnement perturbé. C'est la situation de la 'téléopération directe'. Un fonctionnement que les systèmes équivalents au notre, présents sur le marché, commencent à peine à proposer.

- Une première amélioration consiste à proposer une auto-stabilisation des différents DDL. Ce fonctionnement requiert une architecture logicielle bas-niveau en mesure de porter six boucles d'asservissement contraignant les DDL du robot, en fonction des consignes de l'opérateur : un cap, une profondeur, attitude désirées par exemple. Il est à noter l'impact de cette stratégie sur la relation haptique, l'opérateur pilotant maintenant via une référence en position, ou vitesse. Ce fonctionnement requiert aussi un ensemble de capteurs (centrale inertielle, profondimètre) pour estimer l'*état* du système et ainsi refermer la boucle de rétroaction. Cependant, il faut que ce fonctionnement puisse être engagé – ou non – suivant chaque DDL, l'opérateur pouvant se réserver le pilotage (non-automatique donc) de certains DDL. Cette abstraction de l'étage d'actionnement sera traitée dans le chapitre 3 par l'introduction du répartiteur. Ce répartiteur est d'autant plus important, qu'il doit pouvoir être exploité par les différents robots de la gamme, qu'il s'agisse de la version de base avec 6 moteurs ou dans une version plus évoluée équipée de 12 moteurs. Cette abstraction va permettre également d'exploiter pleinement la redondance structurelle du système. Il est à noter que cette conception d'un niveau d'abstraction supérieur s'inscrit dans une préoccupation de réutilisabilité, puisque la structure de l'étage d'actionnement du système existant est dupliquée, au niveau du skid actionné (version 12 moteurs). Cet ajout de moteurs supplémentaires et de capteurs spécifiques doit être une évolution possible du système. Le contrôle doit pouvoir être adapté à ces modifications structurelles qui impliquent directement le modèle hydrodynamique du robot. La gestion des modes de fonctionnement, la gestion de mission et la gestion de charge utile tout cela doit être pris en compte dans la conception du contrôle.
- L'ouverture de notre système vers les besoins du 'client' implique de proposer des fonctionnalités de plus haut niveau, voire exploitant en ligne les données spécifiques qu'apportent les capteurs additionnels (sédimentomètre, sonar bathymétrique, CTD, fluorimètre, ...). Ainsi, les objectifs du client peuvent être plus finement considérés : suivi de fond, de parois ou de primitives spécifiques, remontée de gradient.... Les besoins du 'client' impliquent de proposer un contrôle modulaire et évolutif.

L'impact de la polyvalence sur le plan de l'automatique n'est pas tant sur les lois de commande elles-mêmes, aussi complexes soient-elles, mais sur la capacité à rendre « transparent » l'actionnement.

2.2.5 Architecture informatique

L'architecture informatique doit également répondre à des exigences de structuration, de composition, de réutilisation et d'évolutivité ; la polyvalence requiert naturellement une architecture ayant ces propriétés.

En termes de modularité, nous nous appuyerons sur le modèle de conception et le modèle d'exécution développés au LIRMM. Le modèle de conception s'appuie sur la décomposition atomique développée et décrite dans les travaux de A. Lasbouygues [40], qui permet de faire le lien entre le monde de l'automatique et le monde de l'informatique, en décomposant les équations issues du monde de l'automatique en éléments de connaissance composables et encapsulables dans les entités de l'architecture logicielle de contrôle. Le modèle d'exécution est celui proposé par le Middleware temps-réel ContrAct que nous utiliserons pour mettre en œuvre notre architecture de contrôle.

Le point sur lequel nous nous focaliserons est relatif à la structuration de l'architecture ; une structuration doit permettre, au regard des différentes applications possibles et des modes de fonctionnement envisagés, de combiner différemment l'ensemble des fonctions de contrôle et celles de mesure, relevant de charges utiles différentes (i.e. capteurs embarqués différents selon l'application). Considérant ces aspects et ceux relatifs à l'architecture matérielle que nous avons brièvement évoqués, nous nous orienterons vers une approche basée service en s'attachant à réifier les contraintes relatives aux ressources, dont les DDL du vecteur.

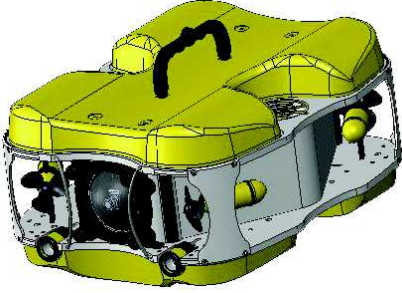

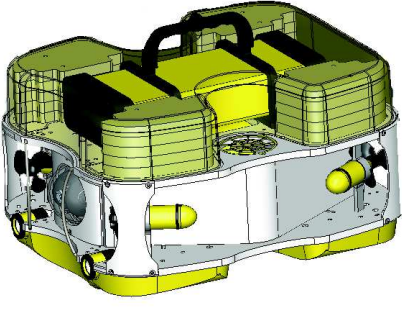
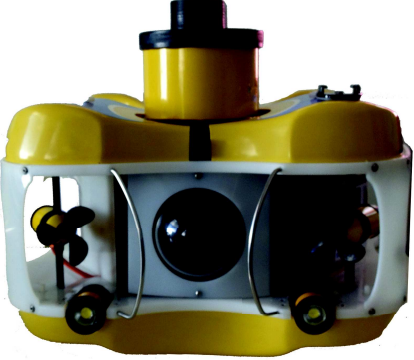
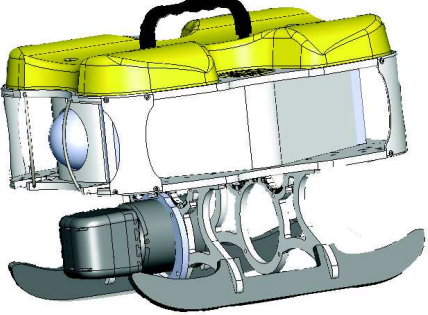

C'est donc selon cette approche que l'architecture logicielle sera structurée, en faisant émerger les modes de fonctionnement, les services et les ressources. Nous aborderons notre structuration basée services dans le chapitre 4, en mettant en exergue tant sa conceptualisation que sa projection sur le middleware temps-réel ContrAct.

2.3 Positionnement – points clefs

- L'objectif de nos travaux est centré sur la polyvalence du vecteur d'exploration, polyvalence que nous avons déclinée sur le plan mécanique, automatique et informatique.
- Les différents prototypes que nous avons réalisés, sur le plan mécanique, durant nos travaux sont décrits Tableau 2.3. Ce tableau met en correspondance les versions conceptuelles que nous avons présentées et les versions prototypes que nous avons réalisées et exploitées à travers nos expérimentations. Ces différentes versions ont pris en considération l'objectif de polyvalence, et la préoccupation de faire émerger une offre industrielle basée sur une gamme de robots.
- Au delà de ces configurations « morphologiques », nous devons décliner la polyvalence au sein de l'architecture de contrôle embarquée, à savoir sur le plan de

l'automatique (chapitre 3) et de l'informatique (chapitre 4). Nous allons à travers le manuscrit nous focaliser sur ces deux aspects.

- Le premier aspect que nous allons considérer est l'abstraction de l'étage d'actionnement et ses propriétés. Nous proposerons un modèle de répartiteur (*control dispatcher*) qui permet un pilotage robuste des propulseurs. Les performances seront validées expérimentalement et théoriquement démontrées dans le cas linéaire.
- Le second aspect concerne une proposition de structuration architecturale 'orientée services' permettant certes d'associer différemment les fonctions selon les modes de fonctionnement mais surtout de réifier les ressources manipulées à travers les différents services, notamment celles relevant de l'abstraction d'actionnement que nous proposons.

Jack de base	
	
Jack de base + Caisson de charge utile (par. ex. pour le CHN)	
	
Jack de base + Skid luge + Caméra acoustique	
	
Jack de base + Skid + Sonar bathymétrique	

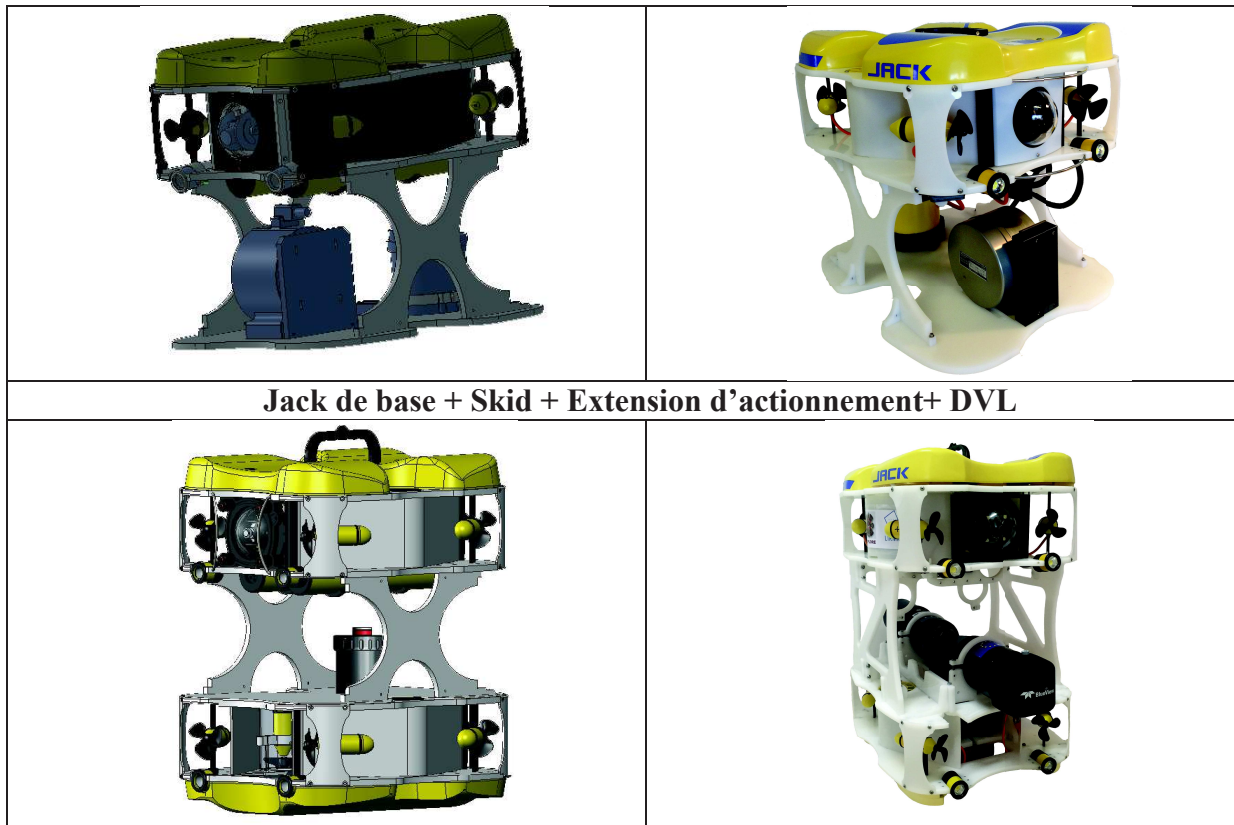


Tableau 2.3 : Prototypes réalisés

Chapitre 3 : Abstraction et propriétés de l'étage d'actionnement : le répartiteur

3.1 Introduction

La démarche que nous suivons sur la question de la polyvalence met clairement en lumière le besoin de modularité pour la conception du système. La diversité des applications implique que le système puisse s'adapter à des besoins d'actionnement différents. Une application d'observation en milieu non contraint (loin de tout obstacle) et non perturbé (pas de courant) requiert un actionnement aux propriétés moins contraignantes que, par exemple, dans le cas applicatif de l'exploration karstique. Ainsi, suivant l'application visée notre système doit pouvoir être doté d'un étage d'actionnement exhibant des propriétés spécifiques, en termes de **puissance propulsive** (pour lutter contre le courant par exemple), de **réactivité** (pour lutter contre des perturbations extérieures) et de **manœuvrabilité** (sous-actionnement / iso-actionnement des DDL), de **robustesse** (vis à vis des paramètres physiques des moteurs) et de **consommation** énergétique.

Dans ce qui suit, nous proposons d'exploiter les propriétés de redondance de notre étage d'actionnement pour traiter de ces questions.

3.2 Notations

Nous présentons tout d'abord les notations qui sont utilisées le long de ce document. Nous abordons aussi la question de la modélisation d'un engin sous-marin. Ces modèles seront ensuite utilisés en simulation. La convention marine, reportée dans [41] est utilisée.

3.2.1 Modèle cinématique

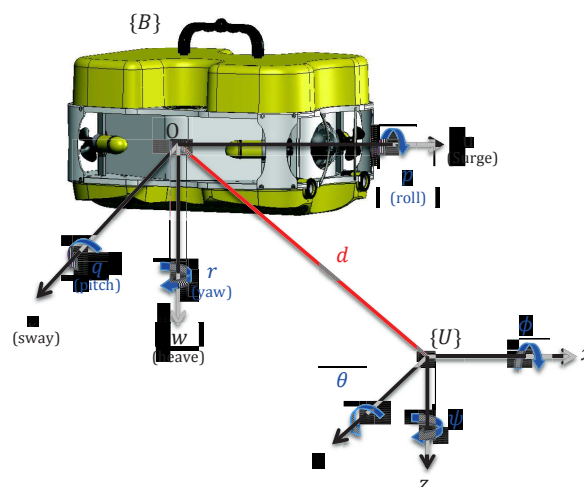


Figure 3-1 : les degrés de liberté du système

Les mouvements sont décrits par rapport à un référentiel inertiel, supposé fixe (galiléen), noté $\{U\}$. Les trois axes de $\{U\}$ sont traditionnellement choisis comme :

- x pointe vers le nord magnétique,
- y pointe vers l'est
- z vers le fond.

Notons que le repère $\{U\}$ est direct. L'attitude de l'engin est décrite suivant la convention roulis-tangage-lacet des angles d'Euler:

- ϕ est l'angle de roulis, il décrit la variation de la gîte du système par rapport au plan horizontal.
- θ est l'angle de tangage, il décrit la variation de l'assiette du système par rapport au plan horizontal.
- ψ est l'angle de lacet et décrit la variation du cap du système par rapport au nord magnétique. Il est aussi appelé cap.

Ainsi est défini le vecteur $\boldsymbol{\eta}_U$ décrivant la situation du système dans le repère absolu $\{U\}$. Il faut noter la singularité intrinsèque à la représentation d'Euler, aussi appelée le blocage de Cardan. En effet, un tangage de $\pm \frac{\pi}{2}$ entraîne une singularité de la définition des deux autres angles ψ et ϕ . De même un roulis de $\pm \frac{\pi}{2}$ ne permet plus de définir de tangage. La représentation d'Euler impose donc que le système évolue avec des angles de tangage et de roulis faibles. Cette représentation singulière peut être évitée avec le formalisme des Quaternions. Les équations (3.1) proposent les deux représentations, avec $\mathbf{P}_U = [x, y, z]^T$ le vecteur de position absolue et \mathbf{Q} le quaternion associé à l'attitude du système.

$$\begin{aligned} \boldsymbol{\eta}_U &= [x, y, z, \phi, \theta, \psi]^T = [\mathbf{P}_U^T, \phi, \theta, \psi]^T \\ \boldsymbol{\eta}_{U,q} &= [\mathbf{P}_U^T, \mathbf{Q}]^T \end{aligned} \quad (3.1)$$

Le repère $\{B\}$ est attaché au véhicule, en un point défini comme le barycentre entre le centre de gravité et le centre de flottabilité (centre volumique) et appelé métacentre. $\{B\}$ se compose de 3 axes orthogonaux décrits comme suit :

- x_B est aligné selon la direction principale de mouvement, il est appelé axe de cavement.
- y_B pointe à tribord et est appelé axe d'embarquée.
- z_B vers le bas, il est appelé axe de pilonnement.

Le repère $\{B\}$ est surtout utilisé pour définir les vitesses linéaires de l'engin dans son propre repère. Ainsi, nous définissons :

- u la vitesse absolue de cavement, suivant x_B , aussi appelée vitesse d'avance.
- v la vitesse absolue d'embarquée, suivant y_B , aussi appelée vitesse latérale.
- w la vitesse absolue de pilonnement, suivant z_B , aussi appelée vitesse monte/baisse.

Les vitesses de rotation autour des axes de $\{B\}$ sont exprimées par les notations suivantes :

- p la vitesse de rotation autour de x_B ,
- q la vitesse de rotation autour de y_B ,
- r la vitesse de rotation autour de z_B ,

Ainsi, nous définissons le vecteur \mathbf{v}_B , exprimant les vitesses de l'engin dans son propre repère. De manière équivalente, la représentation basée sur les quaternions est présentée à l'équation (3.2), avec $\mathbf{V}_B = [u, v, w]^T$ et $\mathbf{W} = [0, p, q, r]$.

$$\begin{aligned} \mathbf{v}_B &= [u, v, w, p, q, r]^T \\ \mathbf{v}_{B,q} &= [\mathbf{V}_B^T, \mathbf{W}] \end{aligned} \quad (3.2)$$

La relation entre la vitesse de l'engin exprimée dans $\{B\}$ et celle exprimée dans $\{U\}$ permet, pour la représentation d'Euler, de définir la matrice $\mathbf{R}(\phi, \theta, \psi)$

$$\mathbf{R}(\phi, \theta, \psi) = \begin{bmatrix} \cos \psi \cos \theta & \cos \psi \sin \theta \sin \phi - \sin \psi \cos \theta & \sin \psi \sin \phi + \cos \psi \sin \theta \cos \phi & 0 & 0 & 0 \\ \sin \psi \cos \theta & \sin \psi \sin \theta \sin \phi + \cos \psi \cos \theta & \sin \psi \sin \theta \cos \phi - \cos \psi \sin \phi & 0 & 0 & 0 \\ -\sin \theta & \cos \theta \sin \phi & \cos \theta \cos \phi & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & \sin \phi \tan \theta & \cos \phi \tan \theta \\ 0 & 0 & 0 & 0 & \cos \phi & -\sin \phi \\ 0 & 0 & 0 & 0 & \frac{\sin \phi}{\cos \theta} & \frac{\cos \phi}{\cos \theta} \end{bmatrix} \quad (3.3)$$

où l'on retrouve les singularités de cette représentation dans les relations entre les vitesses angulaires. Par contre, avec les quaternions, la relation n'est plus singulière. Le modèle cinématique du système est :

$$\begin{aligned} \dot{\boldsymbol{\eta}}_u &= \mathbf{R}(\phi, \theta, \psi) \cdot \mathbf{v}_B \\ \dot{\boldsymbol{\eta}}_{U,q} &= [\dot{\mathbf{P}}_U^T, \dot{\mathbf{Q}}]^T \end{aligned} \quad (3.4)$$

avec $\dot{\mathbf{Q}} = \frac{1}{2} \cdot \mathbf{W} \otimes \mathbf{Q}$ et $\dot{\mathbf{P}}_U^T = \mathbf{Q}^* \otimes [0, \mathbf{V}_B^T] \otimes \mathbf{Q}$, où \otimes est l'opérateur de multiplication de deux quaternions et \mathbf{Q}^* est le quaternion conjugué de \mathbf{Q} .

3.2.2 Modèle dynamique

Le modèle dynamique exprime la relation entre les accélérations, vitesses et attitude du système et les efforts qu'il subit et engendre via son étage d'actionnement. Il est à noter que cette relation s'exprime dans le repère du véhicule $\{B\}$ et n'est pas impactée par les singularités précédemment évoquées.

Notons \mathbf{F}_B le vecteur de dimension (6x1) décrivant les forces et couples résultants de l'étage d'actionnement du système :

$$\mathbf{F}_B = [F_u, F_v, F_w, \Gamma_p, \Gamma_q, \Gamma_r]^T \quad (3.5)$$

Le système immergé subit des efforts hydrodynamiques décrits par les équations des écoulements fluides de Navier-Stokes. Ces efforts, résultant de la vitesse d'écoulement du fluide sur l'engin sont intrinsèquement liés à sa morphologie et il est plus commode de les

exprimer dans le repère de l'engin $\{B\}$. L'approximation au premier ordre de la résolution des équations de Navier-Stokes conduit à définir différents types d'actions hydrodynamiques :

- la masse ajoutée : un système en accélération positive dans un fluide doit fournir aux particules environnantes l'énergie nécessaire à leur mise en mouvement. A l'inverse, lors de la décélération, cette énergie est restituée au système, provoquant au premier ordre, un effet de masse ajoutée. Les abaques de la littérature [42], [43] permettent d'estimer ce paramètre en fonction de la géométrie de l'engin et de la densité du fluide. Il apparaît comme les matrices \mathbf{M}_{add} qui génèrent \mathbf{F}_{add} par la relation :

$$\mathbf{F}_{add} = \mathbf{M}_{add} \cdot \dot{\mathbf{v}}_B \quad (3.6)$$

- la traînée décrit le phénomène d'interaction avec la viscosité du fluide. Elle apparaît comme un terme de frottement visqueux, fonction de la vitesse de l'engin. Elle est fonction de la morphologie que l'engin présente face à l'écoulement [42], [43]. Au premier ordre, sa modélisation donne la relation (3.7).

$$\mathbf{F}_D = \mathbf{D}(\mathbf{v}_B) \cdot \mathbf{v}_B \quad (3.7)$$

- La portance est engendrée par la différence de pression que l'écoulement exerce sur une surface profilée. C'est un effort qui est recherché pour les surfaces de contrôle (gouvernes). Dans notre cas, où le robot est actionné par des propulseurs à hélice, nous ne recherchons pas ce phénomène. De plus, vu les symétries que présente notre robot, ce phénomène est considéré comme négligeable et il ne sera pas considéré dans notre modèle.
- Les couplages hydrodynamiques ont deux origines : i) ils sont dus aux différents profils que le système présente face à l'écoulement, c'est l'effet de la girouette qui, par sa forme s'oriente naturellement dans la direction du vent, et ii) les phénomènes de couplage inertiels qui engendrent les effets de Coriolis, déjà pris en compte dans $\mathbf{C}_{RB}(\mathbf{v}_B)$. Ils sont considérés dans la relation (3.8)

$$\mathbf{F}_C = \mathbf{C}_{cpl}(\mathbf{v}_B) \quad (3.8)$$

- les phénomènes hydrostatiques sont engendrés par un défaut de compensation de la gravité et de la flottabilité. Ils sont considérés par la relation (3.9).

$$\mathbf{F}_f = \mathbf{G}(\boldsymbol{\eta}_U) \quad (3.9)$$

Au final, le modèle dynamique du système sous-marin peut s'écrire, sous forme compacte :

$$\mathbf{F}_B = (\mathbf{M}_{RB} + \mathbf{M}_{add}) \cdot \dot{\mathbf{v}}_B + \mathbf{C}_{RB}(\mathbf{v}_B) \cdot \mathbf{v}_B + \mathbf{D}(\mathbf{v}_B) \cdot \mathbf{v} + \mathbf{C}_{cpl}(\mathbf{v}_B) + \mathbf{G}(\boldsymbol{\eta}_U) \quad (3.10)$$

où les paramètres \mathbf{M}_{RB} et $\mathbf{C}_{RB}(\mathbf{v}_B)$ sont les paramètres du système physique à sec.

De manière plus détaillée, et en ne considérant que les paramètres pertinents pour notre système, le modèle dynamique peut être écrit comme à l'équation (3.11).

$$\begin{aligned}
\mathbf{F}_u &= X_{\dot{u}} \cdot \dot{u} + X_{u,|u|} \cdot u \cdot |u| + X_{w,q} \cdot w \cdot q + X_{v,r} \cdot v \cdot r + X_G(\boldsymbol{\eta}_U) \\
\mathbf{F}_v &= Y_{\dot{v}} \cdot \dot{v} + Y_{v,|v|} \cdot v \cdot |v| + Y_{u,r} \cdot u \cdot r + Y_{w,p} \cdot w \cdot p + Y_G(\boldsymbol{\eta}_U) \\
\mathbf{F}_w &= Z_{\dot{w}} \cdot \dot{w} + X_{w,|w|} \cdot w \cdot |w| + Z_{u,q} \cdot u \cdot q + Z_{v,p} \cdot v \cdot p + Z_G(\boldsymbol{\eta}_U) \\
\mathbf{F}_p &= K_{\dot{p}} \cdot \dot{p} + K_{p,|p|} \cdot p \cdot |p| + K_{q,r} \cdot q \cdot r + K_v \cdot v + K_w \cdot w + K_G(\boldsymbol{\eta}_U) \\
\mathbf{F}_q &= M_{\dot{q}} \cdot \dot{q} + M_{q,|q|} \cdot q \cdot |q| + M_{p,r} \cdot p \cdot r + M_u \cdot u + M_w \cdot w + M_G(\boldsymbol{\eta}_U) \\
\mathbf{F}_r &= N_{\dot{r}} \cdot \dot{r} + N_{r,|r|} \cdot r \cdot |r| + N_{p,q} \cdot p \cdot q + N_u \cdot u + N_v \cdot v + N_G(\boldsymbol{\eta}_U)
\end{aligned} \tag{3.11}$$

Les paramètres hydrodynamiques $[X_{(\cdot)}, Y_{(\cdot)}, Z_{(\cdot)}, K_{(\cdot)}, M_{(\cdot)}, N_{(\cdot)}]$ sont estimés avec une approche classique. De nombreux travaux traitent de l'identification du modèle dynamique non linéaire par exemple dans l'étude des engins autonomes [44]–[48]. [49], [50] traitent plus spécifiquement de la modélisation du robot Jack développé par l'entreprise Ciscrea. Ainsi, le modèle retenu pour notre système sera celui identifié dans [49] et [50].

3.2.3 Simulateurs

Les différents modèles que nous venons d'établir seront utilisés dans les simulations présentées dans la suite du document. Nous avons d'abord développé un simulateur basé sur la représentation d'Euler, considérant de faibles valeurs de roulis et de tangage. L'algorithme est présenté dans le Tableau 3-1.

Initialisation de $\boldsymbol{\eta}_U$ et \mathbf{v}_B Si $(\mathbf{M}_{RB} + \mathbf{M}_{add})$ est inversible alors Tant que ϕ & θ faibles, alors Calculer $\dot{\mathbf{v}}_B$, d'après $\boldsymbol{\eta}_U$, \mathbf{v}_B et \mathbf{F}_B Intégrer $\dot{\mathbf{v}}_B$ Calculer $\mathbf{R}(\phi, \theta, \psi)$ Calculer $\dot{\boldsymbol{\eta}}_u = \mathbf{R}(\phi, \theta, \psi) \cdot \mathbf{v}_B$ Intégrer $\dot{\boldsymbol{\eta}}_u$ Vérifier validité de ϕ et θ . Fin tant que Fin si

Tableau 3-1 : Algorithme reposant sur la représentation d'Euler.

Le simulateur reposant sur le formalisme des quaternions suit l'algorithme dans le Tableau 3-2.

Initialisation de $\boldsymbol{\eta}_{U,q} = [\mathbf{P}_U^T, \mathbf{Q}]^T \mathbf{v}_{B,q}$ Si $(\mathbf{M}_{RB} + \mathbf{M}_{add})$ est inversible alors Tant que ϕ & θ faibles Calculer $\dot{\mathbf{v}}_B$, d'après $\boldsymbol{\eta}_{U,q}$, \mathbf{v}_B et \mathbf{F}_B Intégrer $\dot{\mathbf{v}}_{B,q}$ Calculer $\dot{\boldsymbol{\eta}}_{U,q} = [\dot{\mathbf{P}}_U^T, \dot{\mathbf{Q}}]^T$, avec $\dot{\mathbf{Q}} = \frac{1}{2} \cdot \mathbf{W} \otimes \mathbf{Q}$ et $\dot{\mathbf{P}}_U^T = \mathbf{Q}^* \otimes [0, \mathbf{V}_B] \otimes \mathbf{Q}$ Intégrer $\dot{\mathbf{Q}}$ et normaliser \mathbf{Q} Intégrer $\dot{\mathbf{P}}_U$, composer le nouvel $\boldsymbol{\eta}_{U,q} = [\mathbf{P}_U^T, \mathbf{Q}]^T$ Fin tant que Fin si
--

Tableau 3-2 : Algorithme reposant sur le formalisme des quaternions.

3.3 Définition et réification de l'étage d'actionnement

Nous définissons l'entité logicielle nommée *étage d'actionnement* comme la suite algorithmique qui permet au robot de considérer les consignes issues du *contrôle* pour piloter l'actionnement du système, i.e. gérer les interactions avec l'environnement de façon à engendrer le mouvement désiré. Dans le contexte de la commande NGC, l'étage d'actionnement trouve sa place comme indiqué à la Figure 3-2-a. Nous avons choisi de définir les entrées de l'étage d'actionnement sous la forme d'une force désirée F_B^d représentée dans le repère attaché au robot $\{B\}$. Il produit une force sur l'environnement, elle aussi exprimée dans le repère du robot F_B .

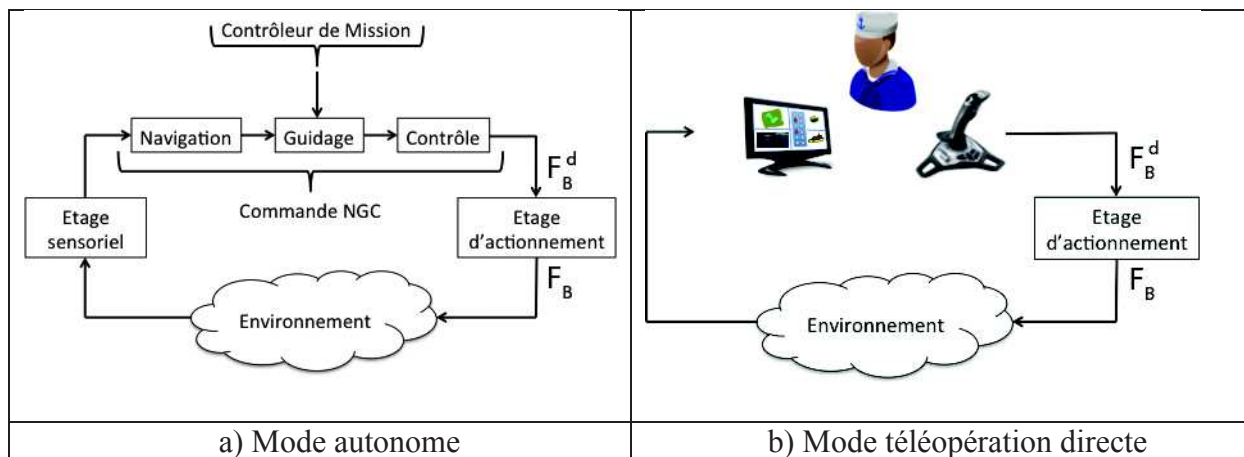


Figure 3-2 : L'étage d'actionnement dans son contexte d'exécution

Dans la situation du mode téléopération directe, la consigne F_B^d provient directement du poste opérateur, comme indiqué à la Figure 3-2-b. Ainsi l'étage d'actionnement est représenté comme sur la Figure 3-3, recevant une consigne F_B^d et produisant une action F_B sur l'environnement.



Figure 3-3 : L'étage d'actionnement

Nombre de qualités du système dépendent des propriétés de l'étage d'actionnement. Pour cela il faut nécessairement passer par une étape de modélisation qui mettra en évidence ces propriétés.

3.4 Modélisation de l'étage d'actionnement

L'étage d'actionnement réalise la transition entre le monde algorithmique et la réalité physique de l'action des actionneurs. Dans notre contexte sous-marin, ces actionneurs sont des propulseurs, que nous définissons comme l'ensemble électronique de commande + moteur électrique + hélice. Le propulseur a ainsi pour rôle de recevoir une consigne (électrique : c_m) et d'engendrer une action mécanique (F_p, Γ_p), comme illustré à la Figure 3-4-a. Par soucis de simplification, nous considérerons dans ce qui suit que le couple de rappel du moteur, Γ_p , est négligeable. Nous verrons plus tard que cette hypothèse est en partie justifiée par le montage vectoriel des moteurs sur la structure du robot. Ainsi nous définissons le modèle du propulseur comme l'entité illustrée à la Figure 3-4-b.

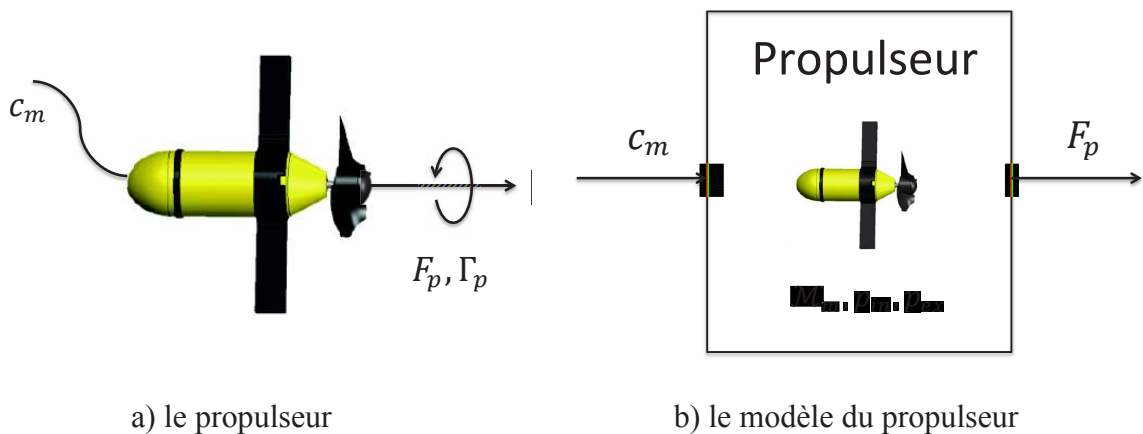


Figure 3-4 : Le propulseur sous-marin.

3.4.1 Modélisation d'un propulseur sous-marin

Un propulseur (sous-marin) est un système électromécanique en mesure d'exercer une force sur l'environnement ambiant (l'eau). Cette force est produite par l'hélice, mise en rotation par le moteur, en interaction avec l'environnement ambiant (l'eau). L'expression de cette force F_p est une fonction f_h de la vitesse de rotation de l'arbre d'hélice (w_h), des caractéristiques de l'hélice (p_h) et des caractéristiques (p_{eau}) du milieu ambiant.

$$F_p = f_h(w_h, p_{eau}, p_h) \quad (3.12)$$

La réaction de l'environnement sur le propulseur induit donc une force de norme F_p , suivant l'axe de hélice et un couple de réaction Γ_p autour de l'axe de l'hélice. Comme stipulé précédemment ce couple sera considéré comme négligeable dans ce qui suit.

La rotation de l'arbre d'hélice est engendrée par celle du moteur. La transmission mécanique hélice-moteur est généralement une relation linéaire (réducteur, de paramètres p_r) entre la vitesse de l'arbre d'hélice (w_h) et celle du moteur (w_m), exprimée par :

$$w_h = f_r(w_m, p_r) \quad (3.13)$$

Hypothèse : L'inertie engendrée par le moteur n'est pas explicitement traitée, mais prise en compte dans l'identification expérimentale faite du propulseur, et attribuée à l'hélice.

Le moteur est de type « courant continu », modélisable comme un système du premier ordre dont les paramètres (p_m) sont connus (constructeur) ou identifiés (voir section 3.4.3.1). Piloté par des signaux PWM, eux mêmes issus du variateur de puissance, on le commande avec une consigne (c_m) dans une gamme $[-c_m^{max}; c_m; c_m^{max}]$.

$$w_m = f_m(c_m, p_m) \quad (3.14)$$

En omettant les dépendances explicites des paramètres du modèle, la relation suivante exprime le **modèle du propulseur** :

$$F_p = f_h(f_r(f_m(c_m, p_m), p_r), p_{eau}, p_h) = \mathcal{M}(c_m) \quad (3.15)$$

Enfin, dernier paramètre du propulseur, la position du point de fixation du moteur au châssis du système est exprimée dans le repère de l'engin $\{B\}$ comme le vecteur de dimension 5 (l'orientation du propulseur autour de son axe de rotation n'est pas un paramètre significatif pour représenter l'action mécanique qu'il induit) :

$$p_x = [x_p, y_p, z_p, \theta_p, \psi_p] \quad (3.16)$$

où les trois premières composantes représentent la position articulaire, θ_p est l'angle d'élévation et ψ_p l'azimut du point de fixation du propulseur au châssis, exprimé dans $\{B\}$. Il faut noter que ce paramètre est extrinsèque au modèle du propulseur, mais sa considération est nécessaire pour combiner l'action de plusieurs propulseurs dans un repère donné.

Nous définissons donc l'objet 'propulseur' comme une entité logicielle comprenant un modèle $\mathcal{M}(c_m)$, des paramètres intrinsèques $p_{in} = [p_m, p_r, p_{eau}, p_h]$, des paramètres extrinsèques $p_{ex} = p_x$, et produisant la force F_p , suivant une entrée c_m , comme illustré à la Figure 3-4-b.

3.4.2 Modélisation de l'action de plusieurs propulseurs

L'ensemble des n propulseurs du système induit une action (force et couple), exprimée dans le repère du robot $\{B\}$ comme le vecteur à 6 composantes :

$$\mathbf{F}_B = [F_u, F_v, F_w, \Gamma_p, \Gamma_q, \Gamma_r]^T \quad (3.17)$$

Elle est induite par l'action des m propulseurs, exprimée sous la forme du vecteur de dimension m défini comme :

$$\mathbf{F}_p = [F_{p,1}, \dots, F_{p,m}]^T \quad (3.18)$$

La relation entre \mathbf{F}_B et \mathbf{F}_p traduit la composition linéaire de l'action de chaque propulseur.

$$\mathbf{F}_B = \begin{bmatrix} F_u = \sum_{i=1}^m \underbrace{-F_{p,i} \cdot \cos \theta_i \cdot \cos \psi_i}_{F_{u,i}} \\ F_v = \sum_{i=1}^m \underbrace{-F_{p,i} \cdot \cos \theta_i \cdot \sin \psi_i}_{F_{v,i}} \\ F_w = \sum_{i=1}^m \underbrace{F_{p,i} \cdot \sin \theta_i \cdot \cos \psi_i}_{F_{w,i}} \\ \Gamma_p = \sum_{i=1}^m F_{v,i} \cdot z_{p,i} + F_{w,i} \cdot y_{p,i} \\ \Gamma_q = \sum_{i=1}^m F_{u,i} \cdot z_{p,i} + F_{w,i} \cdot x_{p,i} \\ \Gamma_r = \sum_{i=1}^m F_{u,i} \cdot y_{p,i} - F_{v,i} \cdot x_{p,i} \end{bmatrix} = \underset{(6 \times m)}{\mathbb{C}} \cdot \mathbf{F}_p \quad (3.19)$$

où \mathbb{C} est une matrice de dimension $(6 \times m)$ que nous appellerons le **concentrateur** que l'on peut aussi trouver dans la littérature sous le nom de « Thruster configuration matrix » [51] où les termes $x_{p,i}, y_{p,i}, z_{p,i}, \theta_i$, et ψ_i désignent la position du propulseur i telle que définie dans (3.16).

La relation inverse nécessite une inversion de la relation (3.19). Si $m > 6$, il y a plusieurs solutions, le système est redondant. Comme nous le verrons plus loin, parmi ces solutions, certaines répondent à des critères de robustesse, de réactivité, de performance énergétique... et confèrent à l'étage d'actionnement les propriétés évoquées en introduction.

La relation inverse s'exprime par la matrice \mathbb{R}_G , de dimension $(m \times 6)$, que nous appellerons le **répartiteur géométrique** (*control dispatcher* en anglais).

$$\mathbf{F}_p = \mathbb{R}_G \cdot \mathbf{F}_B^d \quad (3.20)$$

où \mathbf{F}_B^d est la consigne de l'étage d'actionnement, comme illustré à la Figure 3-2.

L'exploitation des relations précédentes à des fins de contrôle implique d'évaluer les paramètres de l'inverse de la relation (3.15) et de la relation (3.20), i.e. $\widehat{\mathcal{M}}^{-1}(c_m)$ et $\widehat{\mathbb{R}}_G$. La relation (3.21) définit l'étage d'actionnement :

$$\mathbf{F}_B = \mathbb{C} \cdot \mathcal{M} \left(\widehat{\mathcal{M}}^{-1}(\widehat{\mathbb{R}}_G \cdot \mathbf{F}_B^d) \right) \quad (3.21)$$

L'approximation linéaire de la relation (3.15) implique de considérer une caractéristique linéaire des propulseurs et permet d'écrire :

$$\mathbf{F}_p = \mathcal{M}_{lin} \cdot \mathbf{c}_m \quad (3.22)$$

où \mathbf{c}_m est le vecteur de dimension m des consignes moteurs et \mathcal{M}_{lin} est la matrice de dimension $(m \times m)$ des m relations linéaires des caractéristiques des propulseurs (linéarisation de (3.15)). Dans ce cas linéaire l'étage d'actionnement s'écrit :

$$\mathbf{F}_B = \mathbb{C} \cdot \mathcal{M}_{lin} \cdot \widehat{\mathcal{M}}_{lin}^{-1} \cdot \widehat{\mathbb{R}}_G \cdot \mathbf{F}_B^d \quad (3.23)$$

3.4.3 Mise en œuvre expérimentale

Les précédents développements sont appliqués à notre véhicule. Nous commençons par procéder à une identification expérimentale de la caractéristique d'un propulseur. Le concentrateur est ensuite calculé pour le cas du Jack 6. Le répartiteur géométrique en est déduit et testé en situation de téléopération directe. Les effets des zones mortes et de la disparité des caractéristiques moteurs sont mise en évidence.

3.4.3.1 Identification expérimentale de la caractéristique d'un propulseur

La caractérisation des propulseurs est une étape nécessaire pour garantir l'homogénéité de l'étage d'actionnement. Si l'un des moteurs n'a pas les mêmes caractéristiques que les autres, c'est l'ensemble de l'étage d'actionnement qui sera déséquilibré. Nous proposons dans la suite une méthode pour éviter ce problème. Commençons par caractériser un propulseur, à l'aide d'un banc de test (Figure 3-5).

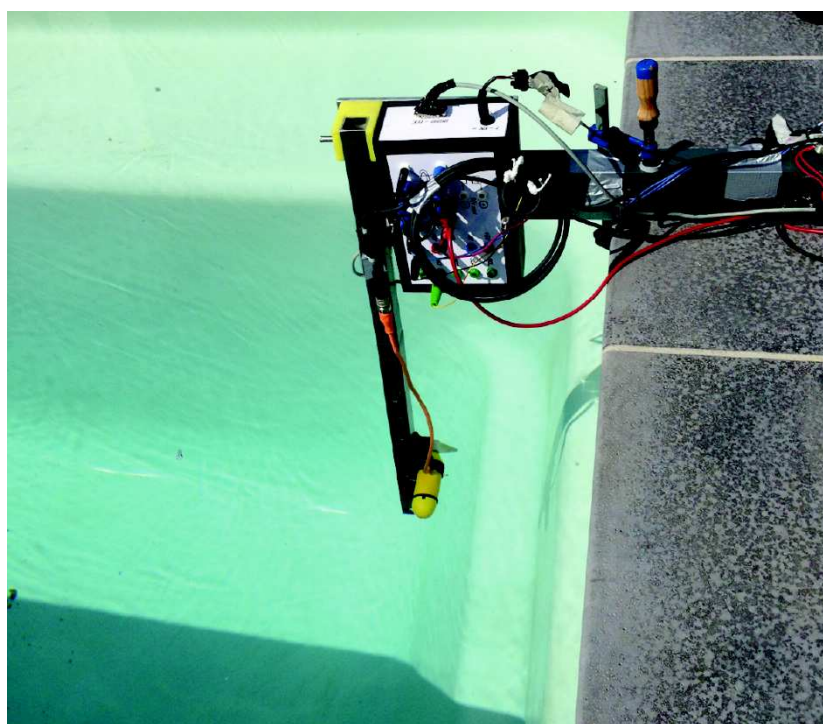


Figure 3-5 : Banc de test moteur

Les mesures ont été réalisées afin d'obtenir une représentation de la force (F_p) en fonction de la consigne (c_m). On peut voir clairement les zones mortes du propulseur dans la Figure 3-6, cette zone morte est la zone où le propulseur ne tourne pas, ou alors que le déplacement

d'eau par l'hélice génère une force insuffisante pour être mesuré par le banc de test. On constate que la force produite par le propulseur n'est pas symétrique lorsque le moteur tourne dans le sens horaire et antihoraire. Ceci est dû à la forme de l'hélice qui d'un point de vue hydrodynamique n'est pas symétrique.

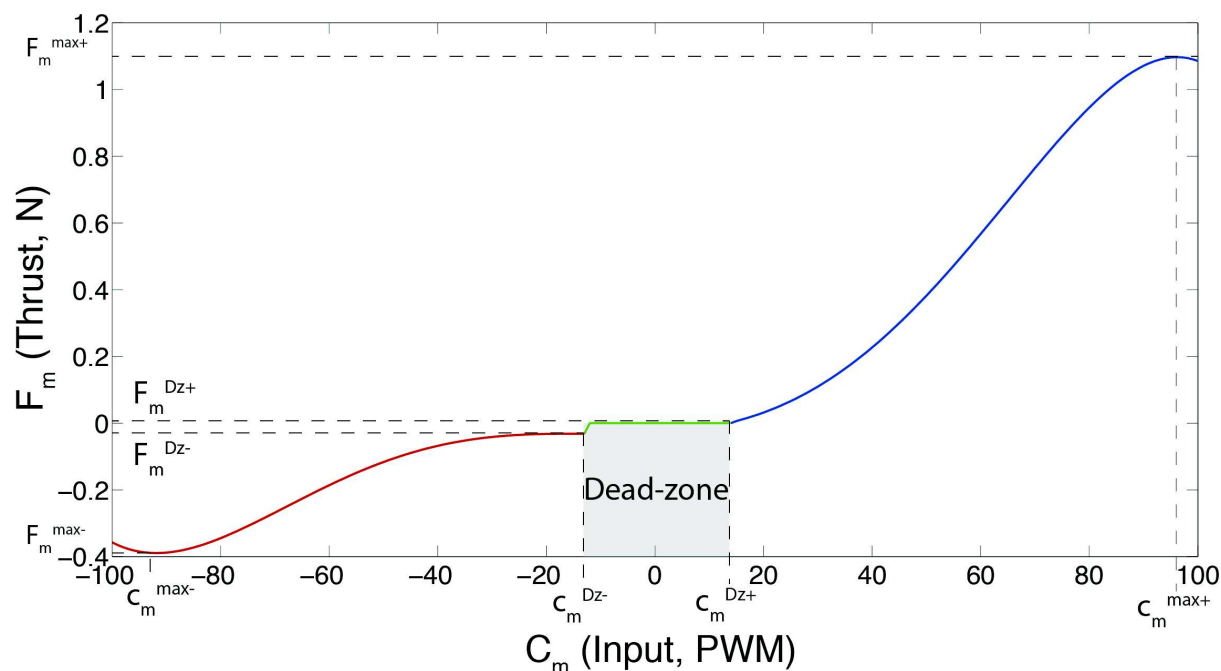


Figure 3-6 : Caractéristique d'un propulseur $F_m = \mathcal{M}(c_m)$.

Des courbes expérimentales de la Figure 3-6, on peut en déduire la fonction polynomiale de $F_p = \mathcal{M}(c_m)$ et son inverse $c_m = \mathcal{M}^{-1}(F_p)$ avec comme coefficient les éléments du Tableau 3-3.

$$\mathcal{M}(c_m) = \begin{cases} \sum_{j=0}^3 a_j \cdot c_m^j, & \text{if } c_m \in [c_m^{\max-}, c_m^{Dz-}] \\ 0, & \text{if } c_m \in]c_m^{Dz-}, c_m^{Dz+}[\\ \sum_{j=0}^3 b_j \cdot c_m^j, & \text{if } c_m \in [c_m^{Dz+}, c_m^{\max+}] \end{cases} \quad (3.24)$$

$$\mathcal{M}^{-1}(F_p) = \begin{cases} \sum_{j=0}^3 d_j \cdot F_p^j, & \text{if } F_p \in [F_p^{max-}, F_p^{Dz-}] \\ 0, & \text{if } F_p \in]F_p^{Dz-}, F_p^{Dz+}[\\ \sum_{j=0}^3 e_j \cdot F_p^j, & \text{if } F_p \in [F_p^{Dz+}, F_p^{max+}] \end{cases} \quad (3.25)$$

j	3	2	1	0
a_j	$-3.2296 \cdot 10^{-5}$	-0.0062	-0.1823	-1.6491
b_j	$-2.4378 \cdot 10^{-5}$	0.0044	-0.1812	2.8060
d_j	0,063	1,28	12,8	-17,39
e_j	1,55	-14,02	54,34	-9,05

Tableau 3-3 : Coefficients de la caractéristique polynomiale produite par un propulseur.

Les caractéristiques précédentes permettent d'identifier les éléments caractéristiques F_p^{max-} et F_p^{max+} , les forces limites que le propulseur est en mesure de générer – pour les consignes c_m^{max-} et c_m^{max+} – et F_p^{Dz-} et F_p^{Dz+} , les forces minimales en limite de zone morte – pour les consignes c_m^{Dz-} et c_m^{Dz+} .

3.4.3.2 Etablissement du concentrateur et du répartiteur géométrique.

Pour cette première approche, nous considérons seulement l'actionnement horizontal du Jack, comme illustré à la Figure 3-7.

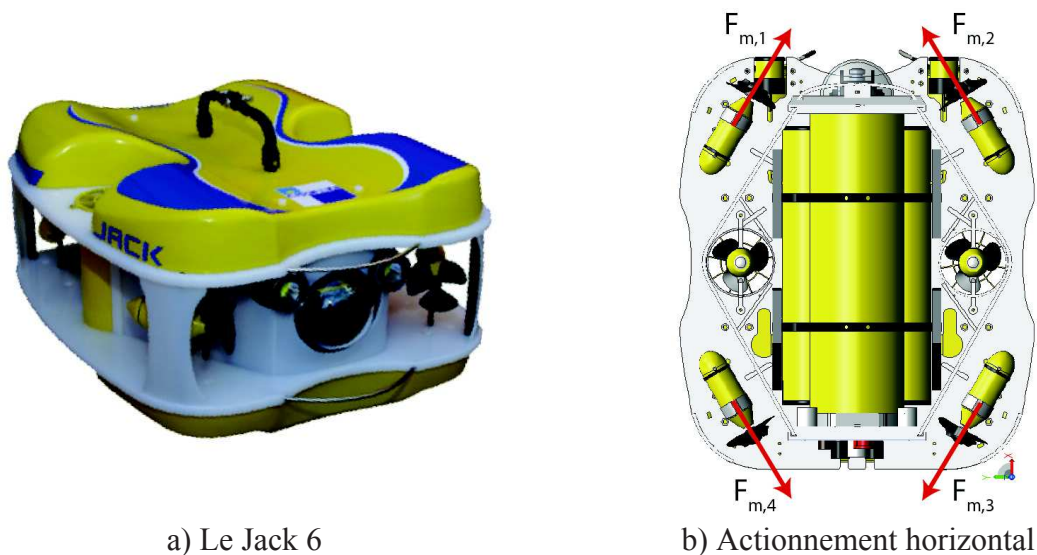


Figure 3-7 : L'actionnement horizontal du Jack

Les caractéristiques mesurées des positions des moteurs par rapport au repère véhicule $\{B\}$ sont reproduites dans le Tableau 3-4.

Id prop.	x_p	y_p	ψ_p
1	.20	-.14	.5236
2	.20	.14	-.5236
3	-.20	.14	3.6652
4	-.20	-.14	-3.6652

Tableau 3-4 : Caractéristique des positions des propulseurs de l'actionnement horizontal.

Le concentrateur horizontal, noté \mathbb{C}_h , est calculé d'après l'équation (3.19)

$$\mathbb{C}_h = \begin{bmatrix} -0.866 & -0.866 & 0.866 & 0.866 \\ -0.5 & 0.5 & 0.5 & -0.5 \\ 0.226 & -0.226 & 0.226 & -0.226 \end{bmatrix} \quad (3.26)$$

Se restreignant à l'étage horizontal, la relation (3.19) devient :

$$\mathbf{F}_{B,h} = \mathbb{C}_h \cdot \mathbf{F}_{p,h} \quad (3.27)$$

où $\mathbf{F}_{B,h} = [F_u, F_v, \Gamma_r]^T$ et $\mathbf{F}_{p,h} = [F_{p,1}, F_{p,2}, F_{p,3}, F_{p,4}]^T$.

La relation inverse de (3.27) est nécessaire au pilotage des actionneurs pour les voir réaliser une action désirée $\mathbf{F}_{B,h}^d$. On constate que \mathbb{C}_h n'est pas carrée et comporte plus de colonnes que de lignes. Le système d'actionnement est donc redondant. L'utilisation de la pseudo-inverse de Moore-Penrose est classiquement utilisée pour identifier une solution inverse qui minimise l'impact énergétique [52], [53]. Dans un premier temps, nous suivons cette approche pour élaborer le répartiteur géométrique qui, réduit à l'étage horizontal, sera noté $\mathbb{R}_{G,h}$:

$$\mathbb{R}_{G,h} = \mathbb{C}_h^T \cdot (\mathbb{C}_h \cdot \mathbb{C}_h^T)^{-1} \quad (3.28)$$

3.4.3.3 Mise en évidence de l'effet des zones mortes.

Nous procédons à une série d'expérimentations en piscine (Figure 3-8), considérant l'asservissement en cap reproduit à l'équation (3.29).

$$\Gamma_r^d = K_P \cdot (\psi_d - \psi) - K_D \cdot \dot{\psi} \quad (3.29)$$

où $\psi_d = 120^\circ$ est le cap désiré pour le système, ψ et $\dot{\psi}$ sont le cap et sa vitesse, mesurés par la centrale inertielle, et $K_P = 50$ et $K_D = 1$ sont les deux gains (positifs) du contrôle proportionnel-dérivé de l'équation (3.29), qui produit le couple désiré, Γ_r^d , pour asservir le cap du robot. Les deux autres forces désirées F_u^d et F_v^d sont choisies nulles pour cette expérimentation. Ainsi, la force désirée produite par le contrôle (3.29) s'écrit : $\mathbf{F}_{B,h}^d =$

$[0,0, \Gamma_r^d]^T$. Nous calculons les consignes à envoyer aux quatre propulseurs grâce à la relation (3.30).

$$\mathbf{c}_m = \mathcal{M}^{-1}(\mathbb{R}_{G,h} \cdot \mathbf{F}_{B,h}^d) \quad (3.30)$$

où \mathcal{M}^{-1} est la relation inverse des caractéristiques inverses des moteurs. Pour cela nous considérons que tous les moteurs ont des caractéristiques identiques à celle identifiée précédemment (3.25).

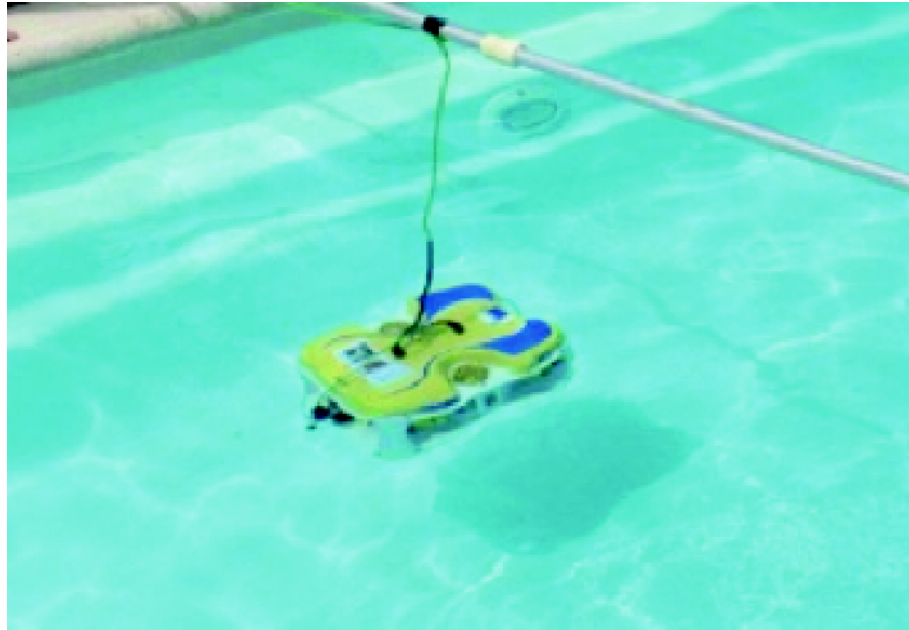


Figure 3-8 : Expérimentation pour mettre en évidence les effets des zones mortes des propulseurs.

Les résultats expérimentaux sont reproduits à la Figure 3-9.

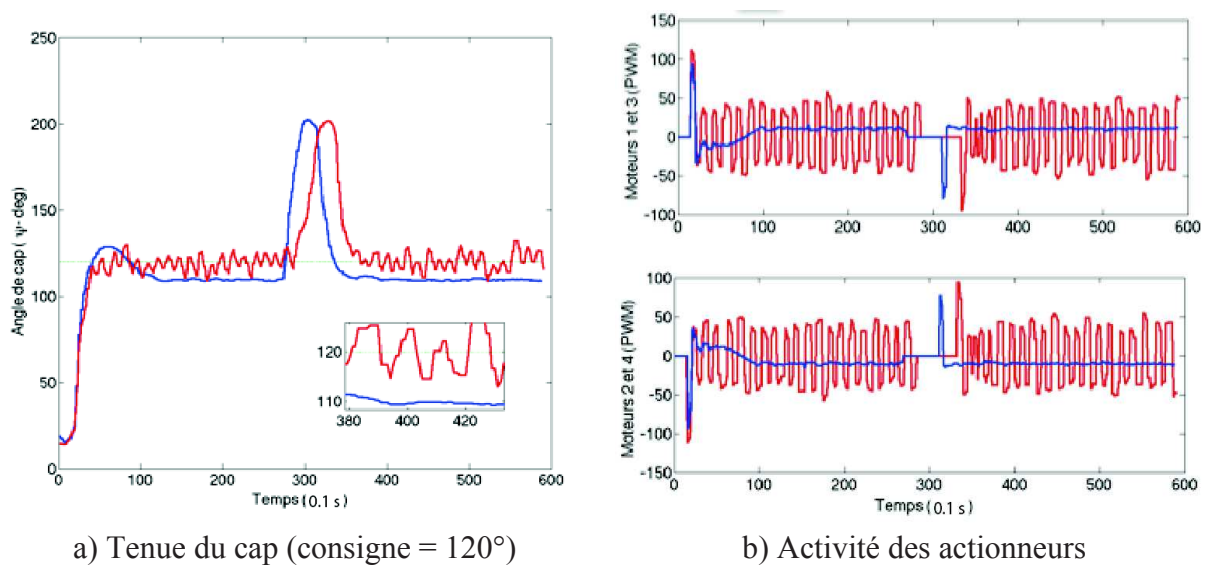


Figure 3-9 : Effet des zones mortes des propulseurs sur la tenue du cap.

Se référant à la Figure 3-9-a, les résultats expérimentaux, suite à l'implémentation du contrôle (3.29) et (3.30), sont reproduits en bleu. Ils indiquent clairement une erreur statique de l'ordre de 10° , ce qui correspond à une erreur qui engendre des consignes actionneurs qui tombent dans leur zone morte. Une perturbation est produite à 30 secondes et les moteurs sont coupés pendant 5 secondes. Nous constatons l'effet de l'asservissement qui ramène le cap dans le voisinage de la consigne, avec une erreur statique.

La manière classique d'éviter de solliciter les actionneurs dans leur zone morte est de réaliser une contraction de la caractéristique de la Figure 3-6, en imposant comme valeurs admissibles pour les consignes moteurs la plage $c_m \in]c_m^{max-}, c_m^{DZ-} [\cup] c_m^{DZ+}, c_m^{max+} [$ (pour notre cas, $c_m \in] -100, -15 [\cup] 15, 100 [$), comme reproduit à la Figure 3-10. La valeurs limites -15 et 15 ont été choisies de telle sorte qu'elles englobent, de manière conservative, la zone morte identifiée.

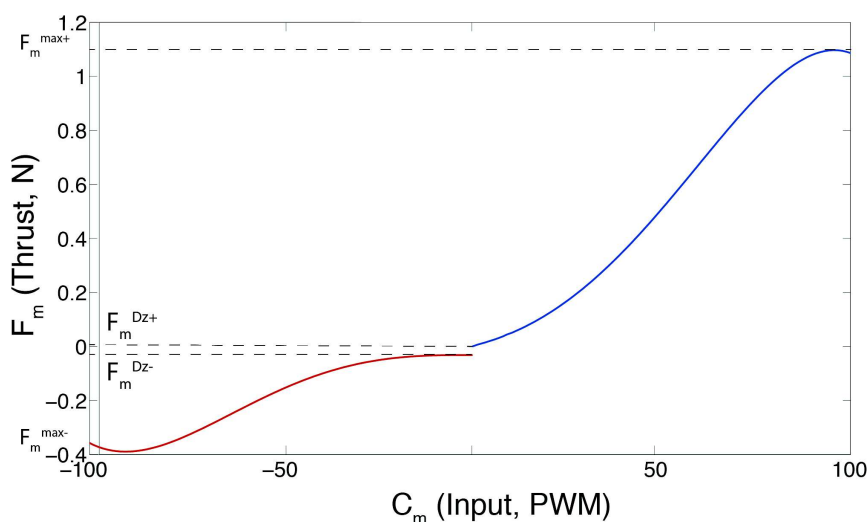


Figure 3-10 : Caractéristique moteur compressée.

L'effet de cette compression annule l'erreur statique et centre la réponse du cap autour de la référence, mais induit de fortes oscillations, comme illustré en rouge à la Figure 3-9-a. Les actionneurs sont alors fortement sollicités, cf. Figure 3-9-b. Une autre conséquence est la perte de réactivité. En effet, autour de la référence de cap (120°), les consignes moteurs alternent entre des valeurs positives et négatives, provoquant une inversion périodique du sens de rotation de l'hélice. Ce type de comportement induit non seulement une usure accélérée des moteurs, mais aussi un manque de réactivité qui est induit par la capacité des moteurs à lutter contre leur propre inertie, et ce dans la pire zone de la caractéristique. La réactivité d'un moteur est à la mesure de la pente de sa caractéristique, faible autour de 0.

Nous verrons dans la suite comment exploiter la redondance de l'étage d'actionnement pour pallier à problème.

3.4.3.4 Mise en évidence de l'effet de la disparité des caractéristiques des propulseurs.

Nous procédons à un nouveau test en piscine pour lequel les consignes propulseurs sont identiques, $c_{m,i} = 80$ PWM pour $i=1,2,3,4$. Théoriquement, grâce au montage vectoriel, si les quatre moteurs horizontaux sont identiques, une telle consigne (commune) ne doit engendrer aucun mouvement. De disposant pas de moyen pour mesurer les vitesses linéaires, nous avons utilisé une séquence vidéo de 10 secondes pour visualiser les écarts de position induite par des mouvements parasites en u et v lors qu'un asservissement en cap est activé (pour n'illustrer que les effets linéaires et pas rotationnels), comme illustré à la Figure 3-11 où les points rouges sont les positions précédentes du vecteur. La Figure 3-12 quant à elle, nous permet de visualiser l'effet parasite sur la rotation du robot.

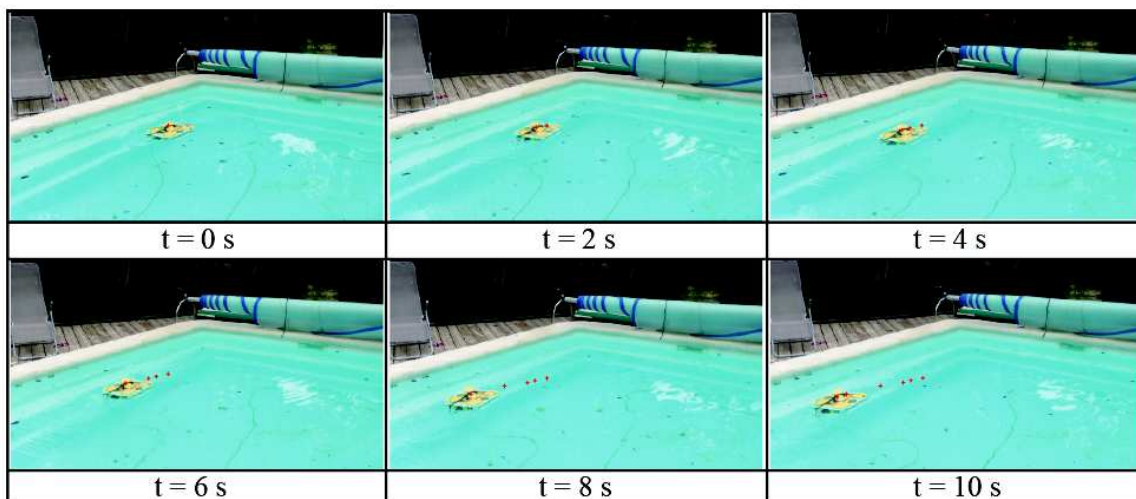


Figure 3-11 : Effet de la disparité de l'étage d'actionnement sur u et v

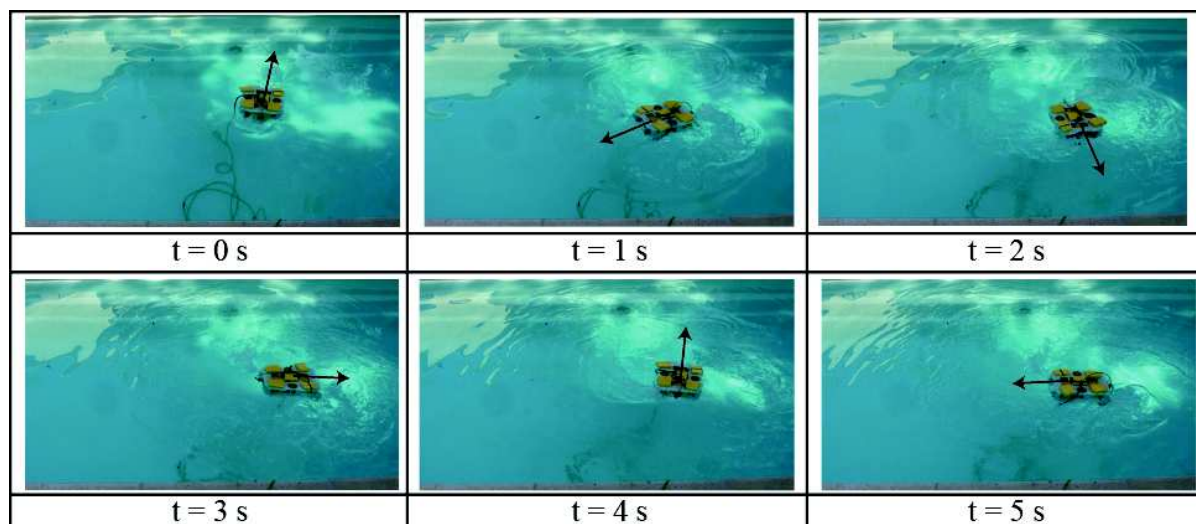


Figure 3-12 : Effet de disparité de l'étage d'actionnement sur ψ

La Figure 3-11 et la Figure 3-12 indiquent clairement que le système ne reste pas en place malgré les consignes communes. Deux causes sont à envisager :

- la disparité des caractéristiques des moteurs qui ne répondent pas identiquement à la même consigne.
- les erreurs d'estimation des paramètres de position des moteurs, qui entachent $\widehat{\mathbb{R}}_{G,h}$ d'erreurs.

Une autre conséquence de ces erreurs est d'engendrer un couplage entre les composantes de $\mathbf{F}_{B,h}^d$. En effet, nous avons constaté qu'une consigne d'avance ($\mathbf{F}_{B,h}^d = [F_u^d, 0, 0]^T$) engendre des mouvements indésirables sur les autres DDL du système. Ceci sera mis en évidence dans la suite, où nous nous attacherons à compenser ces effets de disparités entre les caractéristiques des propulseurs.

3.5 Gestion de la redondance d'actionnement

3.5.1 Etat de l'art

Lors de la dernière décennie, la robotique sous marine a connu un intérêt croissant, avec des applications audacieuses qui nécessitent de pousser plus loin les recherches afin de trouver des solutions originales et efficaces, que ça soit en terme de robustesse, de réactivité et de fiabilité [54].

Les enjeux majeurs actuels sont orientés vers la conception, le management et la gestion de la redondance dans les systèmes d'actionnement. En effet, un grand nombre de véhicules sous-marins ont plus de propulseurs que de degrés de liberté (DDL) [42], [51], [55]–[57].

[55] traite de la redondance structurelle afin d'obtenir de la fiabilité, permettant d'offrir au système une tolérance aux pannes afin de faire face à la perte ou la mise en défaut d'un propulseur. [58] utilise cette tolérance aux pannes pour garantir que l'engin est capable de revenir à station de docking dans le cas où le système perd des capacités d'actionnements. [59] fait appel à une approche heuristique permettant une optimisation de l'efficacité de la poussée des propulseurs.

[60] et [61] traitent des effets de la dynamique des moteurs sur le comportement de l'engin. Les auteurs proposent une série de filtres correctifs pour compenser ces effets et ceux causés par une dégradation des performances du moteur. Ils montrent en outre que ces effets peuvent engendrer, non seulement une perte de performance, mais aussi un cycle limite, en réponse au contrôle pour un positionnement dynamique.

Une bonne caractérisation des propulseurs est nécessaire afin de coller au mieux au modèle réel du propulseur, [62] traite l'identification et du modèle des propulseurs sous-marin. [63] illustre la différence entre un contrôle en boucle fermée et en boucle ouverte et montre ainsi que si le modèle du propulseur est précis, le contrôle en boucle ouverte sera de bonne qualité.

La redondance structurelle du système offre la possibilité au contrôle de réaliser différentes tâches, comme le traite l'approche par fonctions de tâche [64]. Par exemple dans [51], les auteurs utilisent une matrice de configuration de l'état de l'actionnement comme fonction secondaire, à des fins de tolérance aux pannes. Dans [65] et [66] les auteurs proposent d'exploiter la redondance suivant une heuristique qui permet de minimiser l'énergie, malgré

un défaut de compensation des nonlinéarités des moteurs. [56] utilise la logique floue pour faire de la détection de perte de propulseur, la logique floue est également utilisé par [67] pour faire de la compensation dynamique de zone morte pour le maintien de position d'un robot sous-marin.

La gestion de la redondance est une thématique importante de la robotique humanoïde, ou de manipulation [64], [68]. L'approche par fonction de tâche est exploitée dans [69] pour éviter les singularités et la saturation des actionneurs. [70] traite de la priorité qu'on peut attribuer aux tâches secondaires pour des robots manipulateurs.

Dans notre cas, nous utilisons l'approche par tâches secondaires pour ajouter des contraintes supplémentaires à l'étage d'actionnement d'un véhicule sous-marin redondant, afin de supprimer l'effet des zones mortes des propulseurs, de compenser leurs disparités paramétriques, de respecter la saturation et la réactivité du contrôle requise par la tâche.

3.5.2 Construction et analyse des propriétés du répartiteur

La gestion de la redondance de l'étage d'actionnement se fait par le choix du répartiteur \mathbb{R} qui paramètre l'équation (3.31).

$$\mathbf{F}_p = \mathbb{R} \cdot \mathbf{F}^d \quad (3.31)$$

où \mathbf{F}^d est un vecteur, dont la dimension sera discutée plus loin, comprenant la référence \mathbf{F}_B^d . Le choix de la pseudo inverse de Moore-Penrose conduit à l'équation (3.20), reproduite ci-dessous :

$$\mathbf{F}_p = \mathbb{R}_G \cdot \mathbf{F}_B^d \quad (3.32)$$

pour laquelle $\mathbf{F}^d = \mathbf{F}_B^d$ et

$$\mathbb{R}_G = \mathbb{C}^T \cdot (\mathbb{C} \cdot \mathbb{C}^T)^{-1} \leftrightarrow \mathbb{C} \cdot \mathbb{R}_G \cdot \mathbb{C} = \mathbb{C} \quad (3.33)$$

Avec \mathbb{C} défini à l'équation (3.19). Cette solution a l'avantage d'être minimale du point de vue énergétique, mais engendre sur notre système les problèmes évoqués plus haut.

De l'analyse des caractéristiques de \mathbb{C} on déduit certaines propriétés du système. Le nombre de DDL effectivement contraints par l'étage d'actionnement est donné par : $\text{rank}(\mathbb{C})$. Ainsi, quel que soit le nombre de moteurs :

- si $\text{rank}(\mathbb{C}) < \dim(\mathbf{F}_B)$ le système est considéré comme sous-actionné (quel que soit le nombre de moteurs). La stratégie de Guidage-Contrôle idoine s'apparentera à celle dédiée aux systèmes non-holonomes, comme traité dans [71].
- si $\text{rank}(\mathbb{C}) = \dim(\mathbf{F}_B)$, alors les $\dim(\mathbf{F}_B)$ degrés de liberté du système sont contraignables par l'étage d'actionnement. De plus, si $\dim(\mathbf{F}_p) > \dim(\mathbf{F}_B)$, i.e. plus de moteurs que de degrés de liberté, nous sommes en présence d'une redondance structurelle de l'étage d'actionnement.

La redondance du système peut être exploitée en utilisant la projection dans le noyau de \mathbb{C} en utilisant la fonction ci-dessous, introduite par A. Liégeois, en 1977 [72].

$$\mathbb{P}(\mathbb{C}) = (\mathbb{I}_m - \mathbb{R}_G \cdot \mathbb{C}) \quad (3.34)$$

Il démontre que si

$$\mathbf{F}_p = \mathbb{R}_G \cdot \mathbf{F}_B^d + \mathbb{P} \cdot \mathbf{X} \quad (3.35)$$

avec \mathbf{X} un vecteur quelconque de dimension $(m \times 1)$ alors, considérant (3.33), il vient :

$$\mathbf{F}_B = \mathbb{C} \cdot \mathbb{R}_G \cdot \mathbf{F}_B^d + \underbrace{(\mathbb{C} - \mathbb{C} \cdot \mathbb{R}_G \cdot \mathbb{C})}_{=0} \cdot \mathbf{X} = \mathbb{C} \cdot \mathbb{R}_G \cdot \mathbf{F}_B^d \quad (3.36)$$

D'après (3.33), on se retrouve avec la liberté de choisir une consigne additionnelle aux moteurs, sous la forme $\mathbb{P} \cdot \mathbf{X}$, sans toutefois perturber le respect de la référence $\mathbf{F}_B = \mathbb{C} \cdot \mathbb{R}_G \cdot \mathbf{F}_B^d$, pour laquelle $\mathbb{C} \cdot \mathbb{R}_G = \mathbb{I}_n$ si $\text{rank}(\mathbb{C}) = l_{\mathbb{C}}$, où $l_{\mathbb{C}}$ désigne le nombre de lignes de \mathbb{C} (*full row rank*), ou de manière équivalente, que les valeurs singulières de \mathbb{C} soient non nulles [73].

La fonction de tâche primaire est exprimée par \mathbf{F}_B^d et la secondaire par $\mathbb{P} \cdot \mathbf{X}$.

En considérant l'équation (3.35), nous déterminons la relation (3.41) de la redondance de façon compacte entre \mathbf{F}_B^d , \mathbf{F}_p et \mathbf{X} . Posons :

$$\mathbf{X} = \mathbf{M}_m \cdot \mathbf{R}_m \quad (3.37)$$

où \mathbf{M}_m est une matrice $(m \times (m - n))$ et \mathbf{R}_m un vecteur de dimension $((m - n) \times 1)$. Ainsi (3.35) est réécrite comme :

$$\mathbf{F}_p = \mathbb{R}_G \cdot \mathbf{F}_B^d + \mathbb{B} \cdot \mathbf{R}_m \quad (3.38)$$

où $\mathbb{B} = \mathbb{P} \cdot \mathbf{M}_m$. On peut également écrire la relation (3.38) de façon compacte comme ci-après :

$$\mathbf{F}_p = [\mathbb{R}_G \ \mathbb{B}] \cdot \begin{bmatrix} \mathbf{F}_B^d \\ \mathbf{R}_m \end{bmatrix} \quad (3.39)$$

Nous allons maintenant considérer que la matrice \mathbf{M}_m est choisie pour être un élément du noyau de \mathbb{C} , i.e. :

$$\mathbf{M}_m = \text{Ker}(\mathbb{C}) \Leftrightarrow \mathbb{C} \cdot \mathbf{M}_m = \mathbf{0} \quad (3.40)$$

Par conséquent $\mathbb{B} = \mathbf{M}_m$, et l'équation peut être réécrite de la façon suivante :

$$\mathbf{F}_p = \mathbb{R} \cdot \begin{bmatrix} \mathbf{F}_B^d \\ \mathbf{R}_m \end{bmatrix} \quad (3.41)$$

Où $\mathbb{R} = [\mathbb{R}_G \mathbf{M}_m]$, de dimension $(m \times m)$, est appelée **répartiteur** tel qu'il a été introduit à l'équation (3.31). Par analogie avec (3.31), on pose :

$$\mathbf{F}^d = \begin{bmatrix} \mathbf{F}_B^d \\ \mathbf{R}_m \end{bmatrix} \quad (3.42)$$

3.5.3 Exploitation de la redondance.

La redondance structurelle peut être exploitée via le choix de \mathbf{M}_m , dans le respect de (3.40), et de \mathbf{R}_m , vecteur de dimension $((m - n) \times 1)$. Nous voyons dans la suite comment choisir ces composants pour :

- pallier l'effet des zones mortes
- minimiser l'effet de disparité de caractéristique des actionneurs.

3.5.4 Compensation des zones mortes

Dans cette section, nous considérons que les caractéristiques des propulseurs sont identiques et que les matrices \mathbb{C} et \mathbb{R}_G sont connues. Appliquons cette démarche au cas de l'actionnement horizontal du Jack 6 (Figure 6). Le concentrateur \mathbb{C}_h donné à l'équation (3.26) est rappelé ci-dessous :

$$\mathbb{C}_h = \begin{bmatrix} -0.866 & -0.866 & 0.866 & 0.866 \\ -0.5 & 0.5 & 0.5 & -0.5 \\ 0.226 & -0.226 & 0.226 & -0.226 \end{bmatrix} \quad (3.43)$$

$\mathbb{R}_{G,h}$ est ensuite calculée comme la pseudo-inverse de \mathbb{C}_h .

$$\mathbb{R}_{G,h} = \begin{bmatrix} -0.288 & -0.5 & -1.105 \\ -0.288 & 0.5 & 1.105 \\ 0.288 & 0.5 & -1.105 \\ 0.288 & -0.5 & 1.105 \end{bmatrix} \quad (3.44)$$

Comme vu précédemment, \mathbf{M}_m est choisie comme appartenant au noyau de \mathbb{C}_h . Nous remarquons que le choix :

$$\mathbf{M}_{m,h} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \quad (3.45)$$

respecte ce critère, vérifiant que $\mathbb{R}_G \cdot \mathbb{C} \cdot \mathbf{M}_m = \mathbf{0}$. Ce résultat se retrouve dans le fait que la somme des éléments des lignes de \mathbb{C}_h (respectivement les colonnes de $\mathbb{R}_{G,h}$) est nulle. Ceci est une caractéristique du montage vectoriel des propulseurs sur le châssis du véhicule.

Avec ce choix, $\mathbf{R}_m = r_m$ se réduisant à un scalaire puisque $(m - n) = 1$, pour l'actionnement horizontal du Jack 6. Ainsi, on peut écrire que :

$$\mathbf{F}_p = [\mathbb{R}_{G,h} \quad \mathbf{M}_{m,h}] \cdot \begin{bmatrix} \mathbf{F}_B^d \\ r_m \end{bmatrix} \quad (3.46)$$

alors, l'étage d'actionnement réalise la tâche primaire :

$$\mathbf{F}_B = \mathbb{C}_h \cdot \mathbb{R}_G \cdot \mathbf{F}_B^d \quad (3.47)$$

où $\mathbb{C}_h \cdot \mathbb{R}_G = \mathbb{I}_n$ si \mathbb{C}_h est *full row rank*, (vérifié pour 3.43), et ce pour n'importe quelle valeur de r_m .

En d'autres termes cet étage d'actionnement redondant peut être piloté avec un *régime commun* désiré sur chaque moteur, sans modifier le respect de la consigne (3.47).

Ainsi, nous définissons le répartiteur comme :

$$\mathbb{R} = [\mathbb{R}_{G,h} \quad \mathbf{M}_{m,h}] \quad (3.47\text{bis})$$

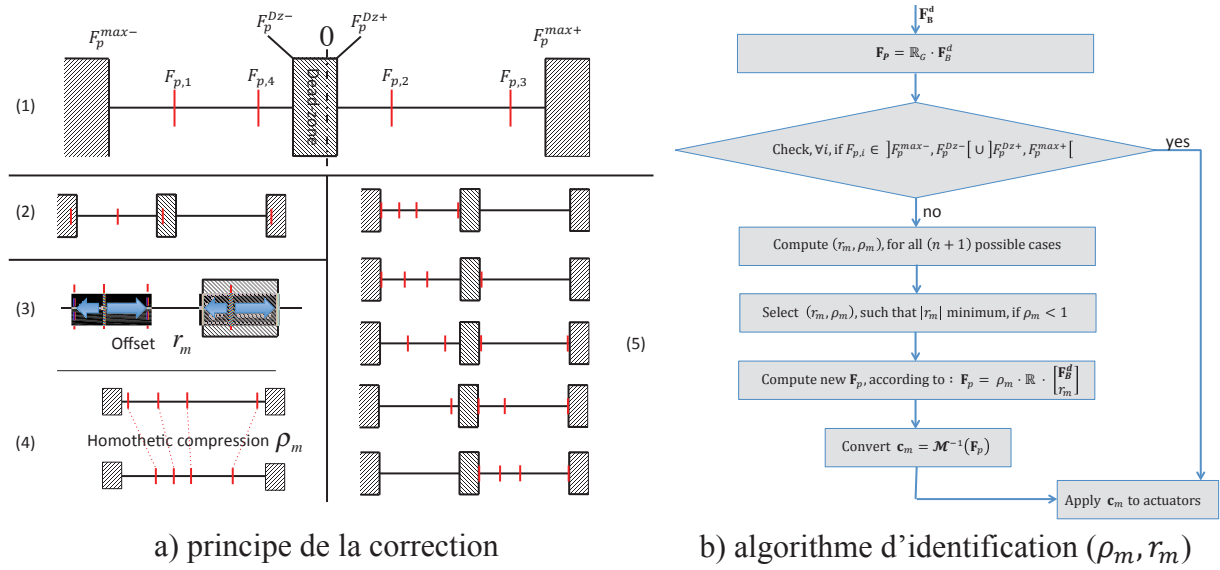
correspondant à l'équation (3.31).

Nous exploitons cette propriété pour piloter le point de fonctionnement des actionneurs, de façon à éviter les zones mortes, mais aussi pour éviter les phénomènes de saturation des actionneurs.

L'algorithme d'identification du régime commun, appliqué à l'actionnement horizontal du Jack est décrit à la Figure 3-13. Le principe est d'identifier un régime moteur commun r_m qui préserve l'ensemble des 4 consignes $c_{m,i}$ de la zone morte, tout en respectant la saturation des actionneurs. Il faut noter que parmi tous les possibles r_m , choisir le minimal minimise les effets sur la consommation énergétique. L'algorithme opère deux transformations sur les consignes $F_{p,i}$, potentiellement problématiques, des propulseurs. Sur la figure Figure 3-13-a(1) les $F_{p,i}$ sont repérés sur un axe présentant des saturations (F_p^{max-} et F_p^{max+}) et la zone morte (F_p^{Dz-} et F_p^{Dz+}). Sur la Figure 3-13-a(2), la distribution des $F_{p,i}$ est clairement infaisable : deux consignes excèdent les limites de saturation et une troisième est dans la zone morte. La première opération consiste à identifier un régime moteur minimal (r_m) qui sorte la consigne de la zone morte. Cette transformation conduit à la distribution de la Figure 3-13-a(3) pour laquelle les consignes sont en dehors de la zone morte. Cependant, l'application du régime commun r_m sur tous les $F_{p,i}$ peut engendrer des consignes qui dépassent les limites de saturation.

La seconde opération consiste à appliquer une compression linéaire homothétique, notée ρ_m , qui préserve les rapports entre les $F_{p,i}$ et les limites de saturation, comme illustré à la Figure 3-13-a(4). Une simple analyse de la complexité montre que dans le cas d'une distribution infaisable de n actionneurs, il existe $(n + 1)$ configurations possibles, i.e. $(n + 1)$ couples de (ρ_m, r_m) acceptables avec un r_m minimal, comme illustré à la Figure 3-13-a(5).

Il faut toutefois remarquer que l'application du coefficient ρ_m modifie la réalisation de la tâche primaire \mathbf{F}_B^d , ce qui n'est pas le cas du régime commun r_m . Du fait de la faible complexité du problème, nous avons établi un algorithme qui inspecte exhaustivement les $(n + 1)$ configurations. La Figure 3-13-b décrit l'algorithme implémenté.



a) principe de la correction

b) algorithme d'identification (ρ_m, r_m)

Figure 3-13 : Principe et algorithme de la compensation des zones mortes.

Revenons sur le fait que l'application de la contraction ρ_m modifie l'application de la consigne \mathbf{F}_B^d . L'analyse de la chaîne complète d'actionnement montre que ce choix conduit à :

$$\mathbf{F}_B = \mathbb{C}_h \cdot \rho_m \cdot \mathbb{R} \cdot \begin{bmatrix} \mathbf{F}_B^d \\ r_m \end{bmatrix} = \rho_m \cdot \mathbb{C}_h \cdot \mathbb{R}_{G,h} \cdot \mathbf{F}_B^d \quad (3.48)$$

Ainsi, dans le cas où \mathbb{C}_h est *full row rank* (nous le vérifions), alors $\mathbb{C}_h \cdot \mathbb{R}_{G,h} = \mathbb{I}_n$ et $\mathbf{F}_B = \rho_m \cdot \mathbf{F}_B^d$ qui résulte donc bien en une transformation homothétique de la consigne \mathbf{F}_B^d .

L'implémentation de cet algorithme sur notre robot fournit les résultats de la Figure 3-14 où la courbe bleue représente le répartiteur statique sans aucune correction de zones mortes, la courbe rouge est le résultat de la compression des zones mortes (Figure 3-9) et la courbe noire est le répartiteur utilisant l'algorithme vu précédemment. On voit clairement sur la courbe noire de la Figure 3-14-a que la tenue du cap est assurée, sans offset ni oscillation. La Figure 3-14-b montre une activité des actionneurs qui se stabilise autour de 20 PWM, qui est la borne que l'on s'est donnée pour éviter la zone morte.

Le fait de rajouter un régime commun à tous les actionneurs a évidemment un impact sur l'énergie consommée. Cependant, on constate que les oscillations des moteurs sont évitées. L'impact du choix du répartiteur sur la consommation énergétique est évalué sur ces expérimentations. Comme nous l'avons vu sur les courbes de la Figure 3-9, l'utilisation du répartiteur géométrique ($\mathbb{R}_{G,h}$) seul avec la caractéristique brute de la Figure 3-6 engendre une réponse sur la tenue du cap qui présente une erreur statique importante, disqualifiant cette méthode. L'utilisation d'une caractéristique compressée, selon la Figure 3-10, permet de centrer l'erreur de cap autour de la consigne. Nous comparons donc l'activité des actionneurs

avec cette méthode par rapport aux résultats obtenus avec le répartiteur augmenté (\mathbb{R}). Nous calculons entre les instants $t_{init} = 1.5s$ et $t_{end} = 25s$ le critère énergétique reporté à l'équation (3.49), basé sur l'activité du moteur 1.

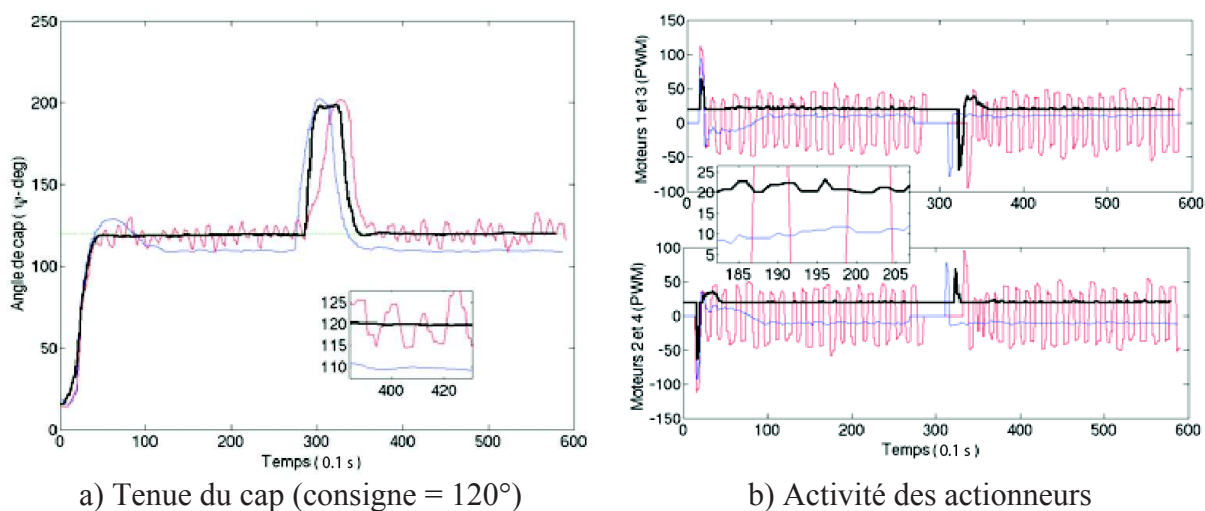


Figure 3-14 : Compensation des effets des zones mortes des propulseurs sur la tenue du cap.

$$\begin{aligned}
 E_{\mathbb{R},G,h} &= \int_{t_{init}}^{t_{end}} |c_{m,1}| \cdot dt \approx 863.6 \\
 E_{\mathbb{R}} &= \int_{t_{init}}^{t_{end}} |c_{m,1}| \cdot dt \approx 526.9
 \end{aligned}
 \tag{3.49}$$

Nous déduisons des précédents calculs que la méthode consistant à considérer un régime commun à tous les moteurs de l'étage d'actionnement réduit la consommation des actionneurs, de par le fait qu'elle fait disparaître les oscillations présentes dans l'autre méthode.

3.5.5 Compensation des phénomènes dus à la disparité de caractéristique des moteurs

Dans la section précédente, nous avons considéré que tous les moteurs étaient identiques, du point de vue de leur caractéristique. Nous tenons à nous attacher dans cette section à présenter une méthode qui permet de lever cette hypothèse forte et rarement vérifiée en pratique.

Afin de présenter cette méthode, nous allons nous attacher dans un premier temps à reproduire les comportements problématiques du système, exposés à la section 3.4.3.4.

3.5.5.1 Simulation des effets de disparité entre actionneurs.

Il est à noter que dans cette section, nous ne considérons pas les incertitudes sur les calculs du concentrateur et du répartiteur, i.e. les paramètres du Tableau 3-5 sont considérés comme justes. Par contre, nous considérons maintenant que les 4 moteurs de l'étage d'actionnement horizontal du Jack répondent à des caractéristiques différentes (inconnues du contrôle), reproduites à la Figure 3-15.

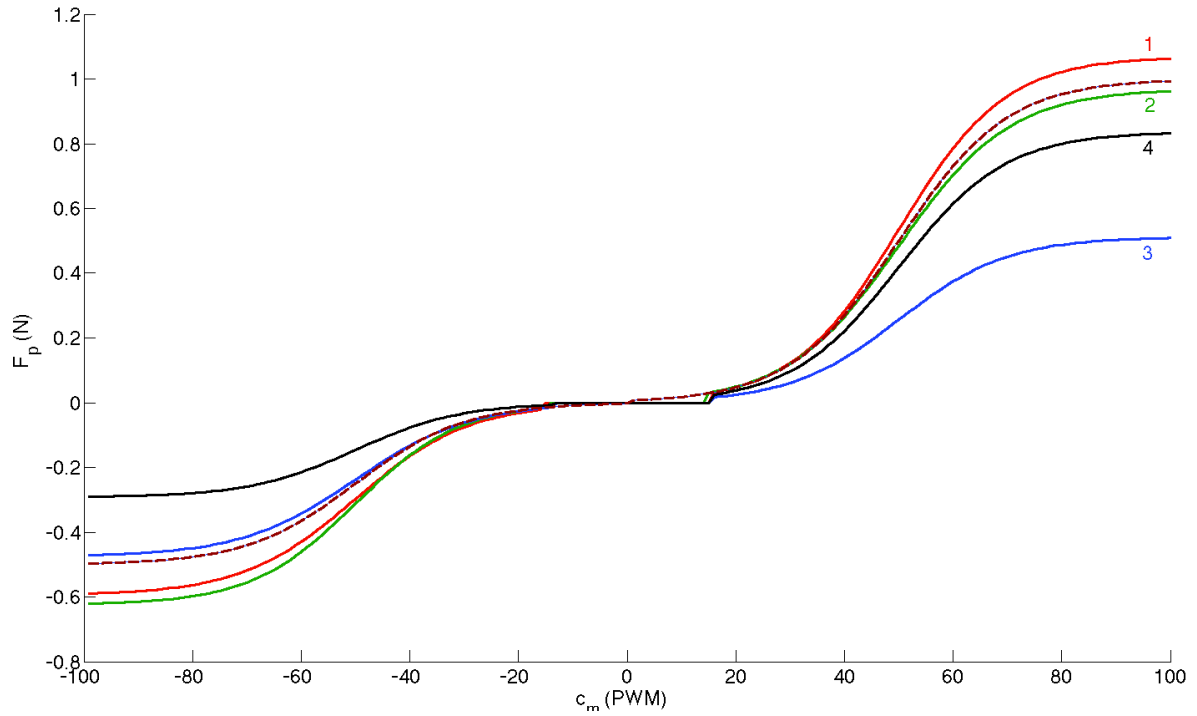


Figure 3-15 : Les caractéristiques simulées des moteurs.

Les courbes de la Figure 3-15 ont été construites d'après le profil générique de l'équation (3.50).

$$F_p = \mathcal{M}(c_m) \begin{cases} \text{si } c_{m,i} > c_m^{Dz+}, \text{ alors } F_{m,i} = \frac{F_{m,i}^{max+}}{2} \cdot (1 + \tanh(\rho_i^+ \cdot (c_{m,i} - 50))) \\ \text{si } c_{m,i} < c_m^{Dz-}, \text{ alors } F_{m,i} = \frac{F_{m,i}^{max-}}{2} \cdot (1 - \tanh(\rho_i^- \cdot (c_{m,i} - 50))) \\ \text{sinon } F_{m,i} = 0 \end{cases} \quad (3.50)$$

Les paramètres des différentes caractéristiques sont définis dans le Tableau 3-5. L'entrée notée 0 correspond à la caractéristique estimée, identique pour tous les propulseurs, et tracée en pointillés oranges sur la Figure 3-15.

i	$F_{m,i}^{max-}$	$F_{m,i}^{max+}$	c_m^{Dz-}	c_m^{Dz+}	ρ_i^-	ρ_i^+
1	-0.5946	1.0688	-15.0000	15.0000	0.0480	0.0513
2	-0.6241	0.9694	-14.2920	14.3580	0.0523	0.0488
3	-0.4753	0.5119	-13.9365	15.7710	0.0475	0.0500
4	-0.2919	0.8371	-13.9080	15.7605	0.0514	0.0510
0	-0.5000	1.0000	0.0000	0.0000	0.0500	0.0500

Tableau 3-5 : Valeur des paramètres des caractéristiques des propulseurs simulés.

Les défauts de comportement mentionnés à la section 3.4.3.4 sont qualitativement reproductibles en simulation.

Nous procédons à une première simulation où les propulseurs reçoivent une consigne identique : $c_{m,i} = c_0 = 28, 50$ et 72 PWM (ces valeurs seront justifiées plus loin). Considérant l'équation (3.21), l'effort résultant produit par l'étage d'actionnement s'exprime selon l'équation (3.51).

$$\mathbf{F}_B = \mathbb{C} \cdot \mathcal{M}((\mathbf{1}_4 \cdot c_0)) \quad (3.51)$$

où $\mathbf{1}_4 = [1,1,1,1]^T$. Sachant que, pour le Jack, $\mathbf{1}_4 \in \ker(\mathbb{C})$, alors une parfaite connaissance des consignes actionneurs ($\widehat{\mathcal{M}}^{-1} = \mathcal{M}^{-1}$) engendrerait un effort \mathbf{F}_B nul. La simulation donne les résultats de la Figure 3-16.

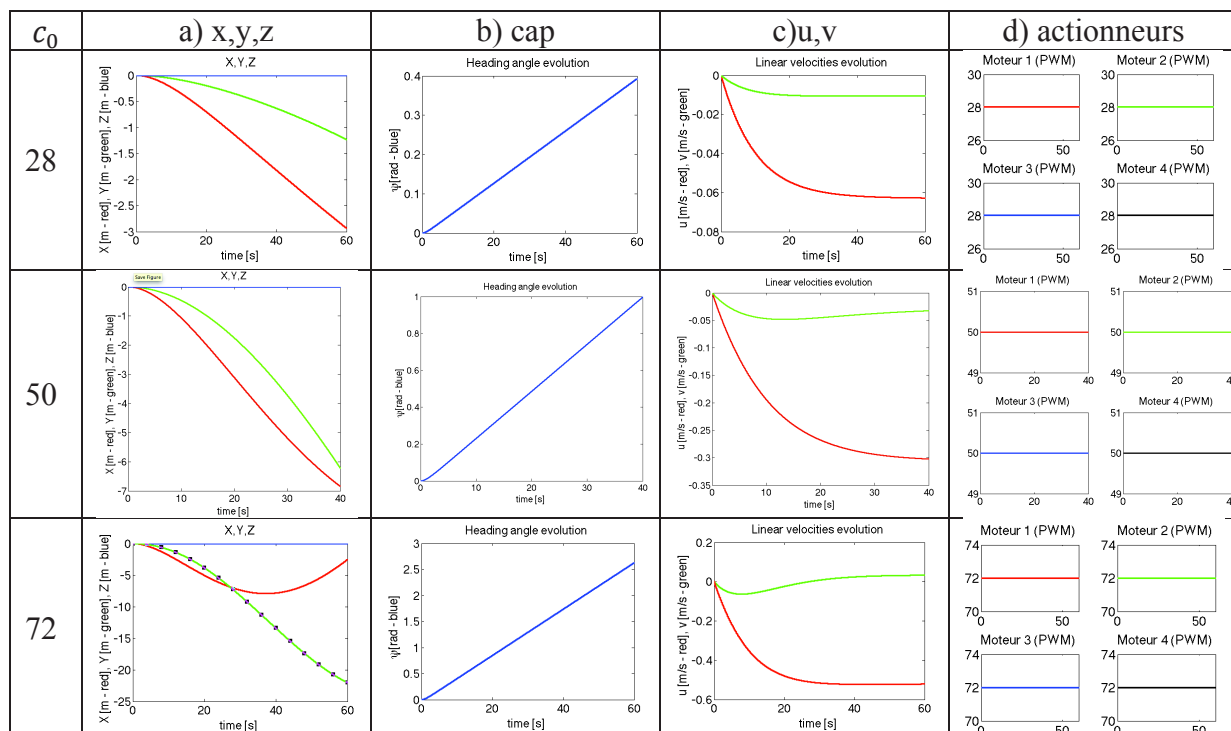


Figure 3-16 : Résultats de simulation pour une consigne identique de 28, 50 et 72 PWM

On peut noter que l'amplitude des variations de position est à la mesure de la variation de cap. Un asservissement en cap améliore grandement les réponses du système, comme illustré à la Figure 3-17, simulation pour laquelle nous avons repris l'asservissement en cap de l'équation (3.29). Compte tenu de l'analyse précédente sur la gestion des zones mortes, un régime commun de $r_m = 0.1N$ a été considéré. L'effort résultant de l'étage d'actionnement est exprimé à l'équation (3.52).

$$\mathbf{F}_B = \mathbb{C} \cdot \mathcal{M} \left(\widehat{\mathcal{M}}^{-1} \left(\mathbb{R} \cdot \begin{bmatrix} \mathbf{F}_B^d \\ r_m \end{bmatrix} \right) \right) \quad (3.52)$$

où $\mathbf{F}_B^d = [0,0,I_r]^T$, avec I_r défini à l'équation (3.19) et \mathbb{R} défini à l'équation (3.47bis). Les résultats de cette simulation sont reproduits à la Figure 3-17.

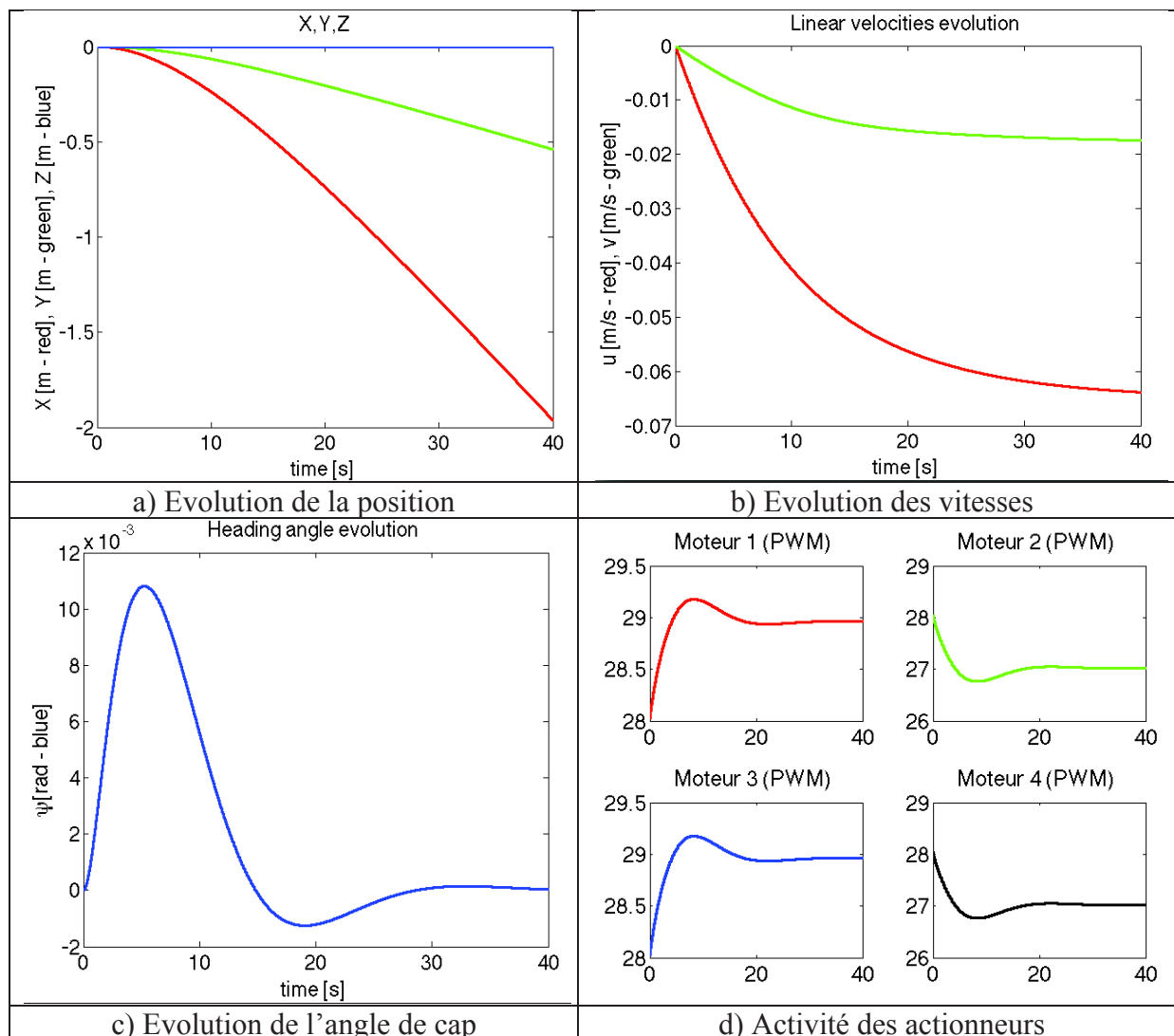


Figure 3-17 : Résultats de simulation avec asservissement du cap.

Les courbes de la Figure 3-17 montrent une bonne tenue du cap, grâce à l’asservissement, Figure 3-17-c. Cependant on est toujours en présence de vitesses linéaires parasites comme le montre la Figure 3-17-b.

Si un asservissement en cap est aisé à obtenir grâce aux mesures d’une centrale inertielle, le contrôle des vitesses linéaires requiert un appareillage bien plus conséquent. L’utilisation d’un Loch doppler permettrait de mesurer ces vitesses, mais requiert de doubler l’étage d’actionnement comme nous l’avons évoqué avant. Ce capteur cher et volumineux s’adapte mal à un système que l’on veut garder léger et de faible coût. Une autre option serait d’utiliser un système de vision qui, par l’analyse du flot optique, permettrait d’estimer le mouvement linéaire du robot. Cependant, cette solution est limitée par la turbidité de l’eau. D’autres solutions sont imaginables, mais requièrent toute un investissement technique et financier qui semble peu adapté à notre vecteur. Dans ce qui suit, nous proposons une méthode d’équilibrage des moteurs qui permet de réduire les effets de disparité entre les caractéristiques des actionneurs.

3.5.5.2 Etablissement des coefficients de correction statique

En analysant les courbes de la Figure 3-17, on voit que les moteurs convergent vers un régime statique qui, s'il est appliqué directement aux moteurs en boucle ouverte, devrait assurer une bonne tenue du cap, sans asservissement. Nous le vérifions en simulation, en appliquant aux moteurs les consignes constantes : $\mathbf{c}_m = [29, 27, 29, 27]^T$. Les résultats de simulation sont ceux de la Figure 3-18. Ils montrent une très faible dérive du cap, à comparer avec la Figure 3-16. Il faut noter que le système est en boucle ouverte. On ne peut donc espérer borner cette dérive, mais l'amélioration est nette.

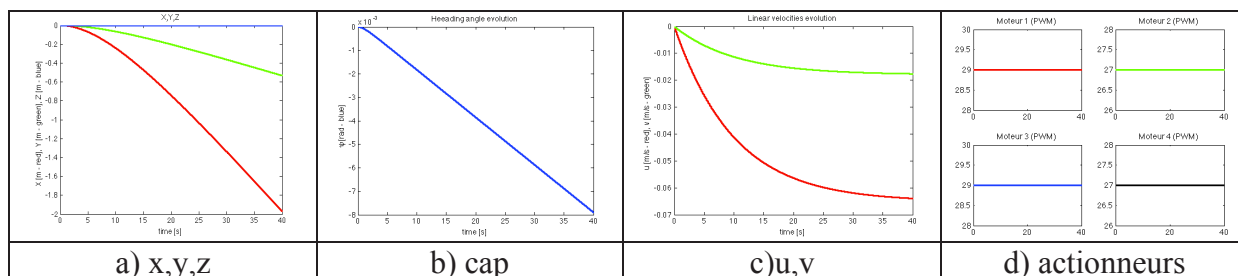


Figure 3-18 : application en boucle ouverte des régimes statiques identifiés sur la Figure 3-17.

Suivant cette logique, nous procédons à une série de simulations pour différents régimes moteurs communs avec un asservissement en cap et un asservissement sur les vitesses linéaires, décrit à l'équation (3.53). Remarquons que ces asservissements des vitesses linéaires sont aisés en simulation. Le passage à la validation expérimentale implique une réflexion sur le choix des capteurs et sur le protocole expérimental, comme nous le verrons plus loin.

$$\mathbf{F}_B^d = \begin{bmatrix} F_u^d = K_{P,u} \cdot (u_d - u) + K_{I,u} \cdot \int_0^t (u_d - u) \cdot dt \\ F_v^d = K_{P,v} \cdot (v_d - v) + K_{I,v} \cdot \int_0^t (v_d - v) \cdot dt \\ \Gamma_r^d = K_{P,r} \cdot (\psi_d - \psi) - K_{D,r} \cdot \dot{\psi} \end{bmatrix} \quad (3.53)$$

où $K_{()}$ sont des gains positifs dont les valeurs sont données dans le Tableau 3-6. Les valeurs désirées u_d , v_d et ψ_d sont prises nulles. Nous appliquons aux actionneurs la consigne de l'équation (3.54).

DDL	K_P	K_I	K_D
u	1	0.1	
v	1	0.1	
ψ	1	 	2

Tableau 3-6 : Valeurs des gains positifs de $\mathbf{K}_{()}$

$$c_m = \widehat{\mathcal{M}}^{-1} \left(\mathbb{R} \cdot \begin{bmatrix} F_B^d \\ r_m \end{bmatrix} \right) \quad (3.54)$$

Nous reportons à la Figure 3-19 les résultats de simulation pour un régime moteur $r_m = 0.1, 0.5$ et 0.9 N.

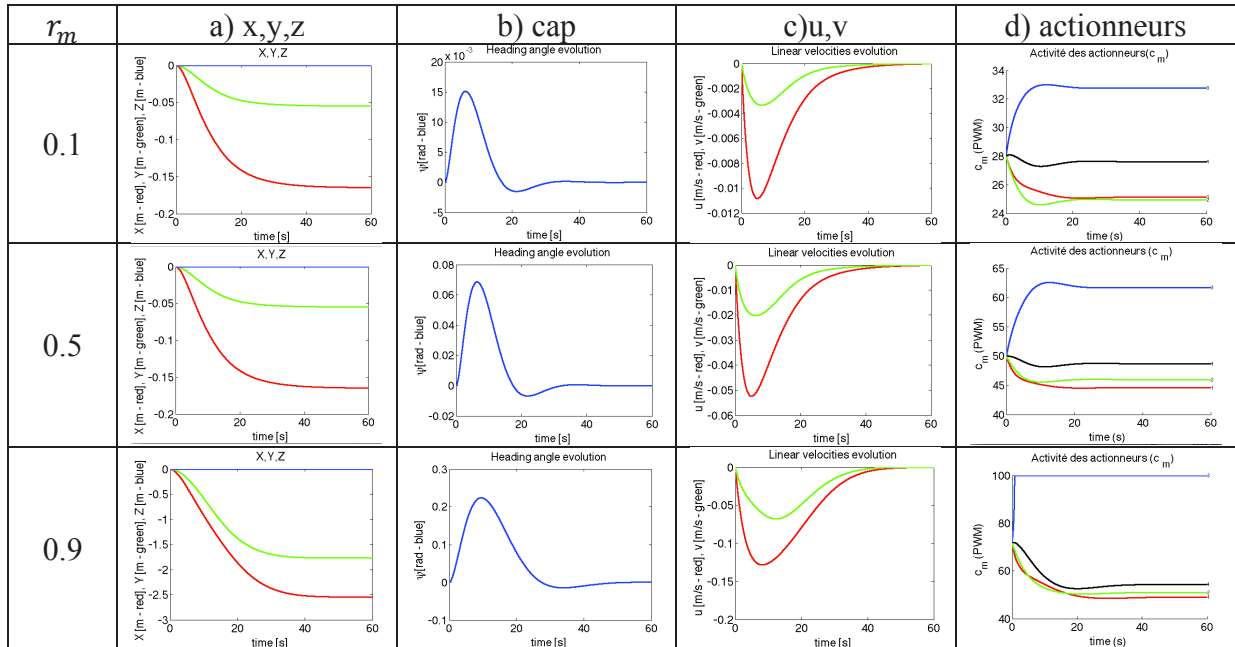


Figure 3-19 : Résultats de l'asservissement autour de régimes moteurs différents.

De l'analyse des résultats de la Figure 3-19, nous identifions les différentes valeurs statiques atteintes par les entrées des moteurs, notées $c_{m,i}^\infty$, reportées dans le Tableau 3-7. La référence r_m est transformée via le modèle inverse estimé, et identique pour tous les moteurs, comme indiqué à l'équation (3.55) :

$$c_0 = \widehat{\mathcal{M}}^{-1}(r_m) \quad (3.55)$$

r_m	0.1	0.5	0.9	
$c_{m,i}^\infty$	1	25.1286	44.6221	49.0556
	2	24.9475	45.9744	51.0056
	3	32.7649	61.7018	100.0000
	4	27.6125	48.6951	54.2800
c_0	0	28.0278	50.0000	71.9722

Tableau 3-7 : Régime statique atteint par les asservissements.

On constate que le régime commun de 0.9 engendre une valeur statique saturée sur la consigne du troisième moteur. La redondance fait que la référence est tout de même tenue. Nous calculons le rapport $\rho_{m,i}$ comme indiqué à l'équation (3.56), et reportons le résultat à la dernière colonne du Tableau 3-8.

$$\rho_{m,i} = \frac{c_{m,i}^{\infty}}{c_0} \quad (3.56)$$

r_m		0.1	0.5	0.9
$\rho_{m,i}$	1	0.8966	0.8924	0.6816
	2	0.8901	0.9195	0.7087
	3	1.1690	1.2340	1.3894
	4	0.9852	0.9739	0.7542
	0	1.0000	1.0000	1.0000

Tableau 3-8 : Les coefficients de correction pour un régime commun r_m

Nous procédons de même sur une plage de régime commun de -0.5 N à 1 N, par pas de 0.1N, la consigne actionneur c_0 étant calculée selon l'équation (3.54). A chaque itération nous relevons les différents régimes statiques $c_{m,i}^{\infty}$ et calculons les $\rho_{m,i}$. Nous reproduisons à la Figure 3-20 l'évolution des différents coefficients de correction, calculés selon (3.56).

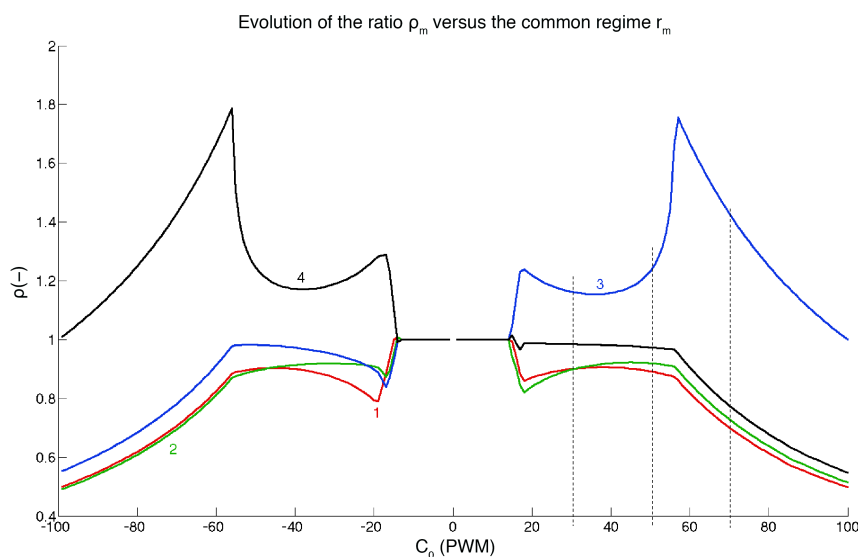


Figure 3-20 : Evolution des coefficients de correction en fonction des régimes moteurs communs.

La Figure 3-20 montre les différentes valeurs des coefficients de correction pour différents régimes moteurs. On remarque l'effet des zones mortes (approximativement entre -20 et 20 PWM). Les discontinuités des courbes autour des -60 et 60 PWM correspondent à des régimes communs qui engendrent des saturations. Ceci est confirmé par la Figure 3-21 qui montre l'évolution des régimes statiques atteints par les moteurs.

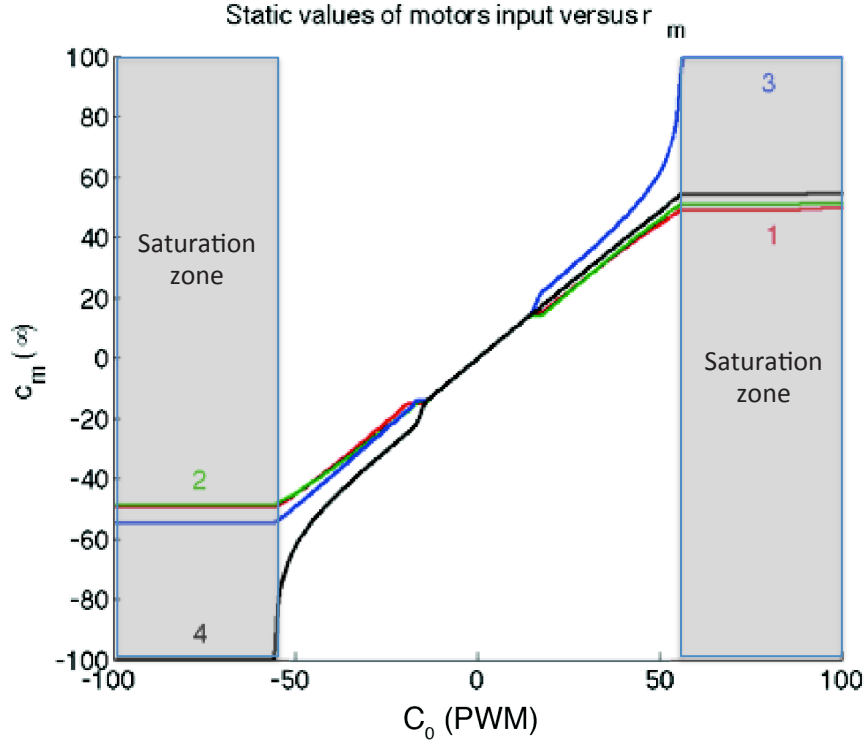


Figure 3-21 : Régime statique atteint par les moteurs, pour différents régimes communs

On note $\boldsymbol{\rho}_m(\mathbf{c}_0) = [\rho_{m,1}(c_0), \rho_{m,2}(c_0), \rho_{m,3}(c_0), \rho_{m,4}(c_0)]^T$ le vecteur composé des coefficients de correction des moteurs en fonction du régime commun r_m , avec $\mathbf{c}_0 = [c_0, c_0, c_0, c_0]^T$. On note aussi la matrice de correction $\mathbb{Q}(\boldsymbol{\rho}_m)$, définie comme à l'équation (3.57).

$$\mathbb{Q}(\boldsymbol{\rho}_m(c_0)) = \mathbb{Q}(c_0) = \begin{bmatrix} \rho_{m,1}(c_0) & 0 & 0 & 0 \\ 0 & \rho_{m,2}(c_0) & 0 & 0 \\ 0 & 0 & \rho_{m,3}(c_0) & 0 \\ 0 & 0 & 0 & \rho_{m,4}(c_0) \end{bmatrix} \quad (3.57)$$

Dans le cas général on considère (3.57bis) et (3.57ter), qui s'écrivent :

$$\boldsymbol{\rho}_m(\mathbf{c}_m) = \begin{bmatrix} \rho_{m,1}(c_{m,1}) \\ \rho_{m,2}(c_{m,2}) \\ \rho_{m,3}(c_{m,3}) \\ \rho_{m,4}(c_{m,4}) \end{bmatrix} \quad (3.57\text{bis})$$

$$\mathbb{Q}(\mathbf{c}_m) = \begin{bmatrix} \rho_{m,1}(c_{m,1}) & 0 & 0 & 0 \\ 0 & \rho_{m,2}(c_{m,2}) & 0 & 0 \\ 0 & 0 & \rho_{m,3}(c_{m,3}) & 0 \\ 0 & 0 & 0 & \rho_{m,4}(c_{m,4}) \end{bmatrix} \quad (3.57\text{ter})$$

On utilise la matrice de correction, en boucle ouverte, suivant l'équation (3.58).

$$\mathbf{c}_m = \mathbb{Q}(\mathbf{c}_0) \cdot \widehat{\mathcal{M}}^{-1} \left(\mathbb{R} \cdot \begin{bmatrix} 0 \\ 0 \\ 0 \\ \widehat{\mathcal{M}}(\mathbf{c}_0) \end{bmatrix} \right) \quad (3.58)$$

Dans la Figure 3-22, nous comparons la réponse du système en boucle ouverte à l'application d'une consigne de régime $c_0 = 28.0278$, 50 et 71.9722 PWM, avec l'utilisation de la matrice de correction $\mathbb{Q}(\mathbf{c}_0)$. Ces consignes de régime correspondent à des régimes communs (théoriques) de $r_m = 0.1$, 0.5 et 0.9 N. Les résultats de la Figure 3-22 sont à comparer à ceux de la Figure 3-16.

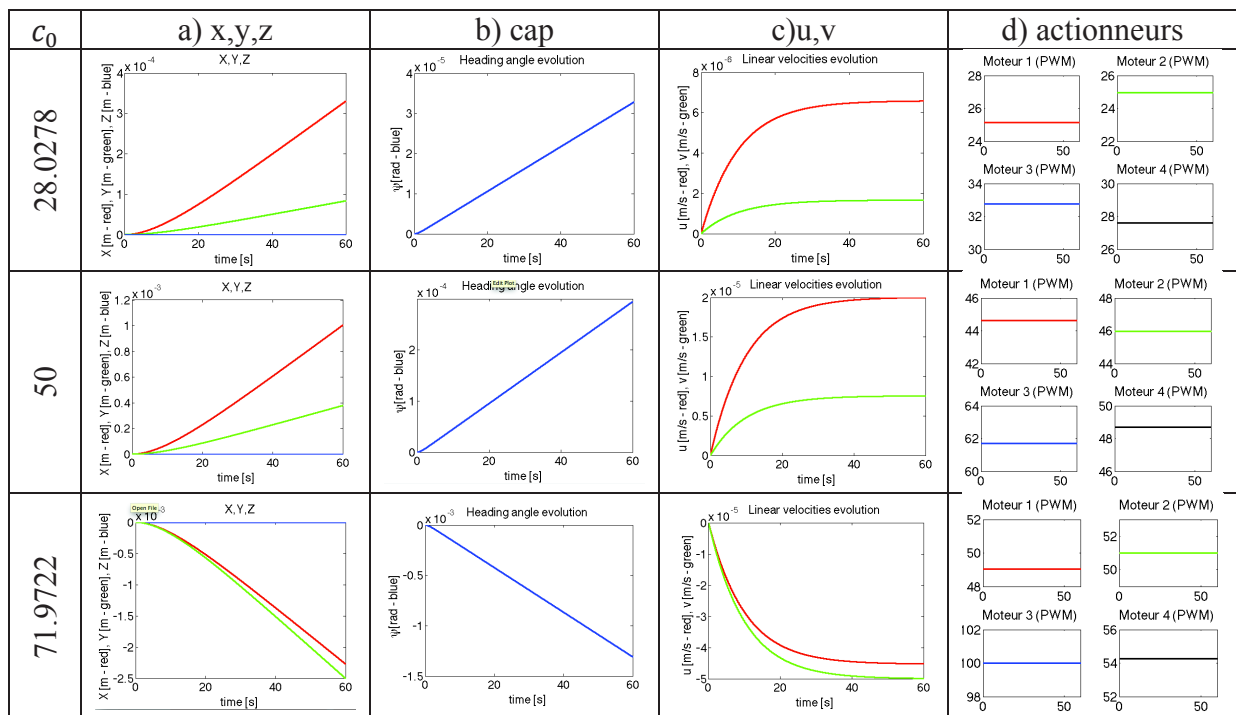


Figure 3-22 : Réponse du système à différents régimes communs, avec correction.

La Figure 3-22 montre une nette amélioration de la réponse du système à différents régimes communs. On note cependant que l'évolution des erreurs n'est pas bornée, ce qui ne peut être atteignable en boucle ouverte. Pour l'instant nous avons considéré la consigne \mathbf{F}_B^d

comme nulle. Nous verrons plus loin ce que cette correction apporte dans la situation où \mathbf{F}_B^d est non nulle.

3.5.5.3 Utilisation des coefficients de correction

Les développements précédents peuvent être mis à profit pour réaliser différents objectifs. Nous en présentons deux dans ce qui suit.

3.5.5.3.1 Caractérisation des moteurs

Imaginons que la caractéristique d'un des moteurs est connue. On considère, pour l'exemple, que nous connaissons la caractéristique du moteur 1, décrite à la Figure 3-15. Les caractéristiques des autres propulseurs sont inconnues. Nous utilisons la redondance du système pour réaliser les tâches de régulation précédemment décrites en imposant le régime du moteur 1. Nous considérons le contrôle de l'équation (3.53) et appliquons la transformation en consignes actionneurs décrite à l'équation (3.59)

$$\mathbf{c}_m = \widehat{\mathcal{M}}^{-1} \left(\mathbb{R} \cdot \begin{bmatrix} \mathbf{F}_B^d \\ f_1(c_0) \end{bmatrix} \right) \quad (3.59)$$

où \mathbb{R} est décrit à l'équation (3.47bis) et $f_1(c_0)$ est une fonction qui permet d'imposer au moteur 1 la consigne désirée tout en préservant la régulation. On choisit $f_1(c_0)$ comme décrit à l'équation (3.60).

$$f_1(c_0) = \mathcal{M}_1(c_0) - \mathbb{R}_G(1, :) \cdot \mathbf{F}_B^d \quad (3.60)$$

où $\mathbb{R}_G(1, :)$ représente la première ligne du répartiteur géométrique \mathbb{R}_G . Il faut noter que l'hypothèse de la connaissance de la caractéristique du moteur 1 fait que $\widehat{\mathcal{M}}_1 = \mathcal{M}_1$. Ainsi, considérant (3.59) et la composition de \mathbb{R}_G décrite en (3.47bis), il vient pour le moteur 1 (tâche secondaire):

$$c_{m,1} = \mathcal{M}_1^{-1}(\mathcal{M}_1(c_0)) = c_0 \quad (3.61)$$

sous réserve que \mathcal{M}_1 soit une fonction inversible. L'asservissement (3.53) permet, quant à lui, de garantir que le système se stabilise autour de vitesses (rotationnelle et linéaires) nulles. Puisque le degré de redondance du système est de 1, on peut dire que dans une telle situation de stabilité (une fois le régime statique atteint), on a : $F_{m,1} = F_{m,2} = F_{m,3} = F_{m,4}$. Ainsi, connaissant la caractéristique du moteur 1 et les régimes statiques que les moteurs ont atteint grâce aux asservissements, on peut dresser (partiellement) la caractéristique de tous les moteurs.

On applique ce principe à notre simulateur, pour différentes valeurs de c_0 (consigne du moteur 1) et on relève les différentes valeurs de $F_{m,1}$ en fonction des $c_{m,i}^\infty$. Les courbes sont montrées à la Figure 3-23.

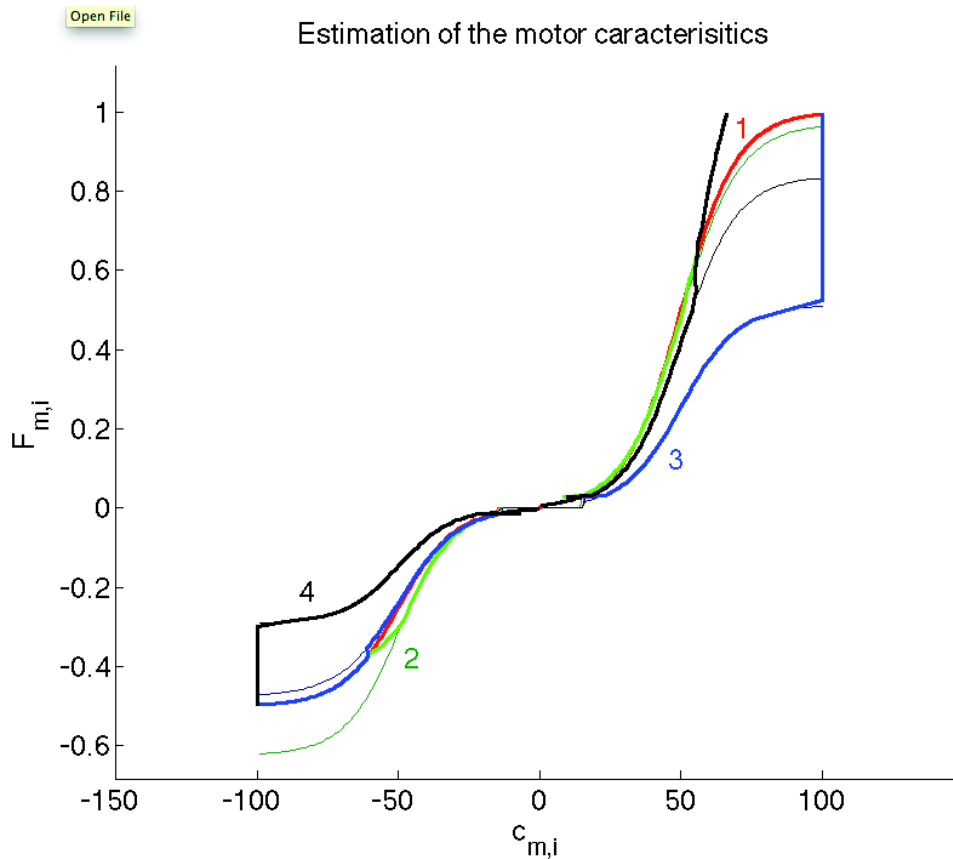


Figure 3-23 : Estimation des caractéristiques des 4 moteurs, connaissant celle du moteur 1.

La Figure 3-24-a indique les régimes statiques atteints par les moteurs. On met en évidence les zones où les moteurs saturent, et celles correspondant aux zones-mortes. La Figure 3-24-b montre la norme de l'erreur en fin d'asservissement.

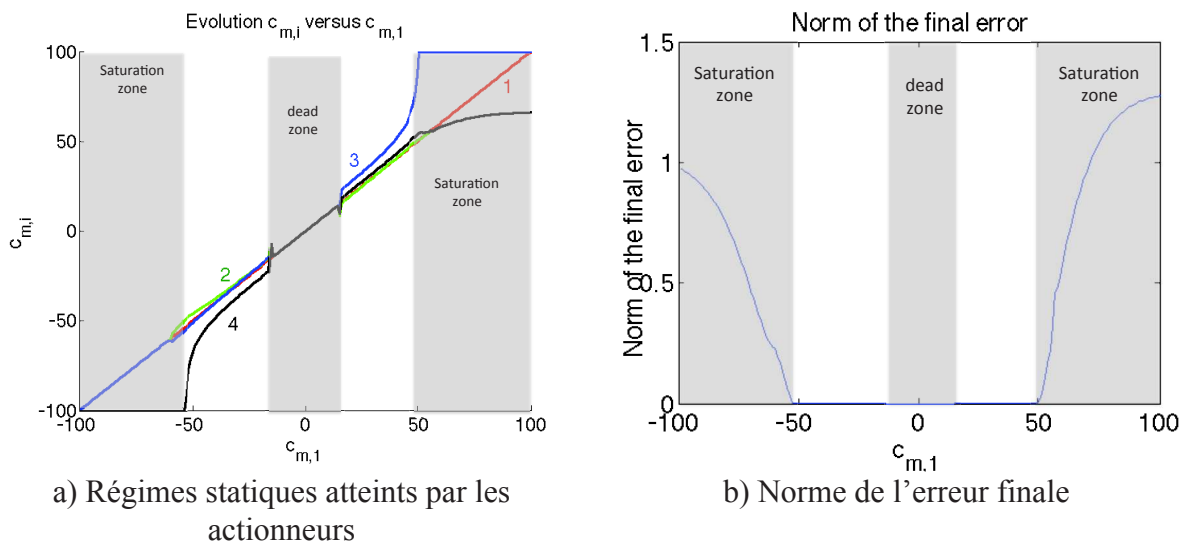


Figure 3-24 : Repérage des zones d'effets des saturations et des zones mortes.

Ainsi, dans les zones où ni les zones mortes ni les saturations ne sont présentes, on peut effectuer une identification (partielle) des caractéristiques des moteurs, comme indiqué à la Figure 3-25.

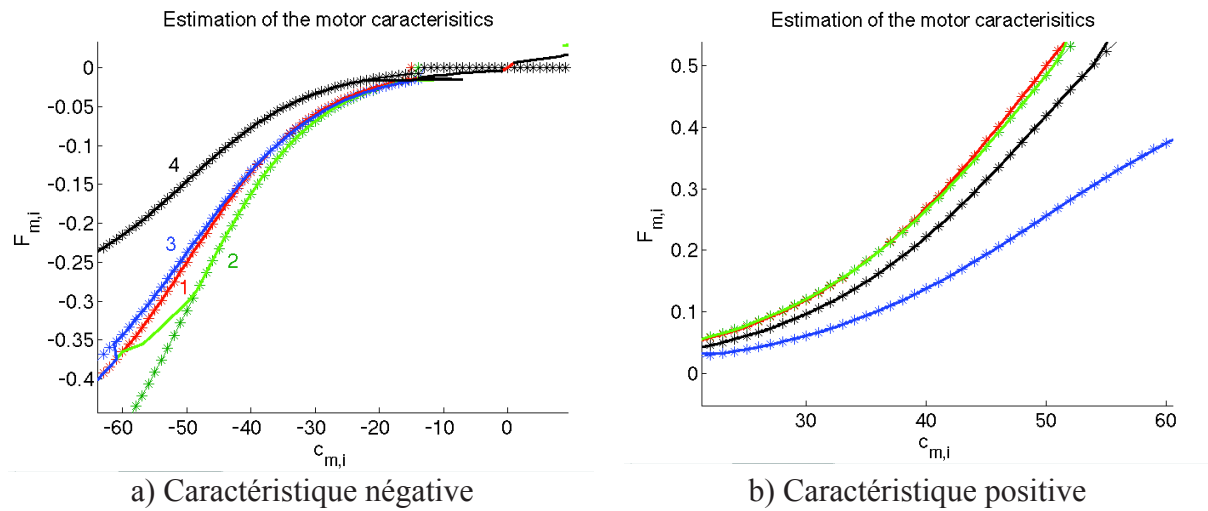


Figure 3-25 : Comparaison des caractéristiques réelles (*) et des caractéristiques estimées, en dehors des zones mortes ou saturées.

3.5.5.3.2 Découplage de l'action des propulseurs

La disparité des caractéristiques des actionneurs a aussi pour effet de 'coupler' la relation (3.62), comme le montrent les courbes de la Figure 3-26 pour lesquelles $\mathbf{F}_B^d = [1,0,0]^T$, $\mathbf{F}_B^d = [0,1,0]^T$ et $\mathbf{F}_B^d = [0,0,1]^T$.

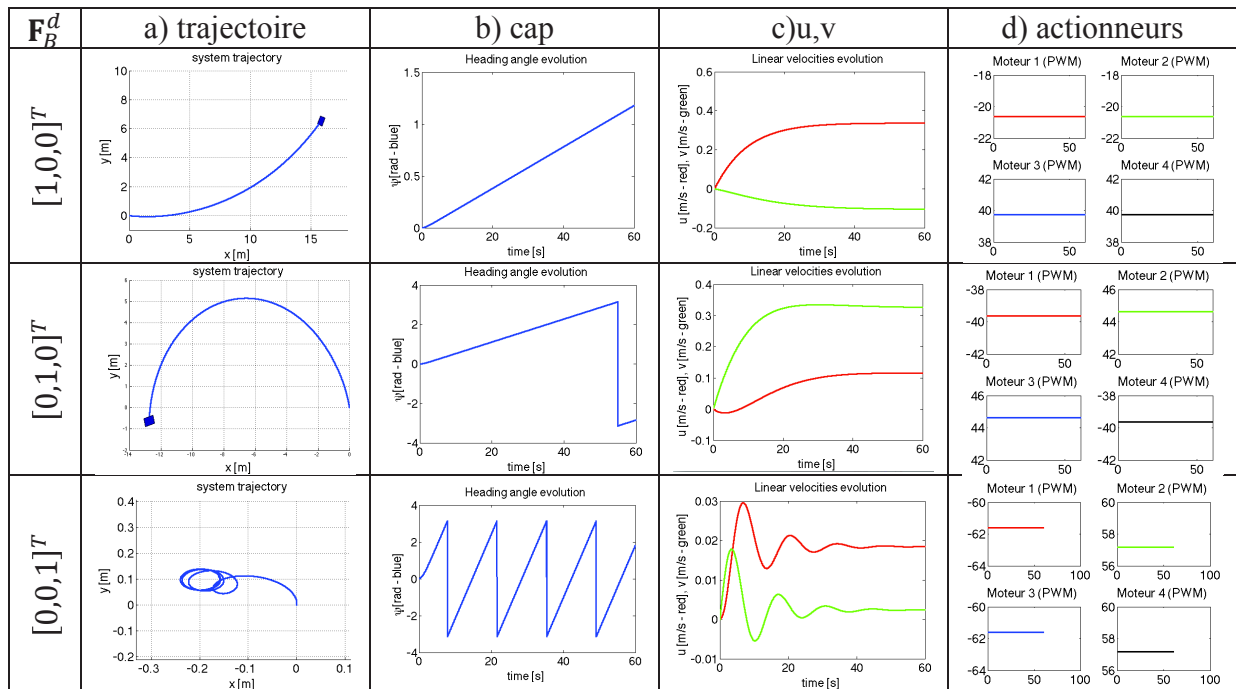


Figure 3-26 : Effet de couplage de la disparité des caractéristiques des actionneurs, $\mathbf{F}_B^d = [1, 0, 0]^T$.

$$\mathbf{F}_B = \mathbb{C} \cdot \mathcal{M} \left(\widehat{\mathcal{M}}^{-1} (\mathbb{R}_G \cdot \mathbf{F}_B^d) \right) \quad (3.62)$$

La Figure 3-26 montre que la réponse du système se fait suivant d'autres DDL que celui qui est sollicité par $\mathbf{F}_B^d = [1,0,0]^T$, i.e. suivant la direction u . Cet effet de couplage des DDL se comprend aisément si on considère l'approximation linéaire :

$$\mathbf{F}_B = \mathbb{C} \cdot \mathcal{M}_{lin} \cdot \widehat{\mathcal{M}}_{ln}^{-1} \cdot \mathbb{R}_G \cdot \mathbf{F}_B^d \quad (3.63)$$

En effet, même si les matrices \mathbb{C} et \mathbb{R}_G sont parfaitement connues, que \mathbb{C} est *full row rank* (impliquant que $\mathbb{C} \cdot \mathbb{R}_G = \mathbb{I}_n$) et que la matrice résultante $\mathcal{M}_{lin} \cdot \widehat{\mathcal{M}}_{ln}^{-1}$ est diagonale (dans le cas linéaire), rien ne garantit que la matrice résultante $\mathbb{C} \cdot \mathcal{M}_{lin} \cdot \widehat{\mathcal{M}}_{ln}^{-1} \cdot \mathbb{R}_G$ soit diagonale (condition de découplage). Cette condition de découplage est en effet une qualité recherchée, ne serait-ce que pour la téléopération.

Revenons maintenant au cadre général, et considérons la matrice de correction \mathbb{Q} , telle qu'elle a été établie à l'équation (3.58), avec le contrôle en boucle ouverte de l'équation (3.64).

$$\mathbf{c}_m = \mathbb{Q}(\mathbf{c}_{m,0}) \cdot \mathbf{c}_{m,0} \quad (3.64)$$

où $\mathbf{c}_{m,0} = \widehat{\mathcal{M}}^{-1} \left(\mathbb{R} \cdot \begin{bmatrix} \mathbf{F}_B^d \\ r_m \end{bmatrix} \right)$.

L'application du contrôle (3.64) a pour effet de découpler la relation (3.62), comme le montrent les résultats de simulation de la Figure 3-27.

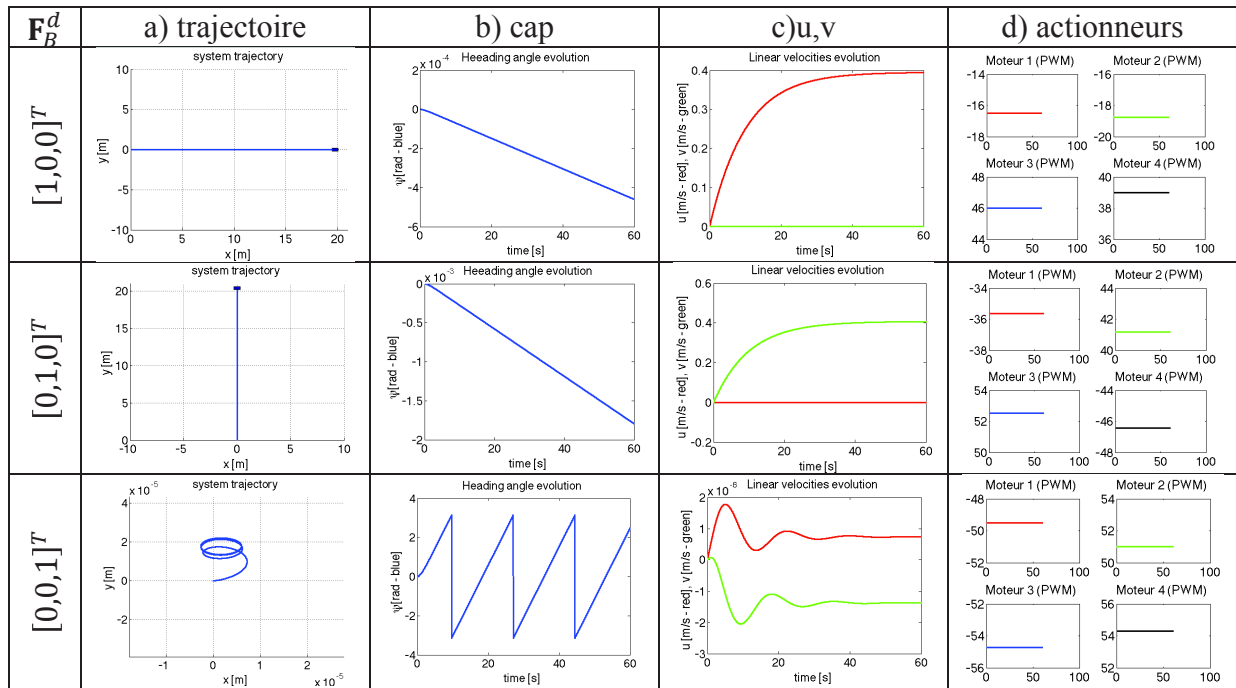


Figure 3-27 : Effet de découplage du système grâce au contrôle en boucle ouverte (54).

Notons à nouveau que ce contrôle est en boucle ouverte et donc que la dérive est toujours présente, mais grandement diminuée. Cette action de découplage peut être expliquée dans le cas linéaire.

3.5.5.3.3 Analyse

On peut expliquer le phénomène de découplage en se restreignant à l'approche linéaire. Elle contraint les caractéristiques actionneurs \mathcal{M}_{lin} à être linéaires. Dans ce cas, \mathcal{M}_{lin} est une matrice ($m \times m$) diagonale, constituée des pentes constantes des caractéristiques de chaque moteur. L'équation (3.52) peut alors s'écrire :

$$\mathbf{F}_B = \mathbb{C} \cdot \mathcal{M}_{lin} \cdot \widehat{\mathcal{M}}_{lin}^{-1} \cdot \mathbb{R} \cdot \begin{bmatrix} \mathbf{F}_B^d \\ r_m \end{bmatrix} \quad (3.66)$$

Considérons pour cette situation les caractéristiques linéaires des actionneurs reproduites à la Figure 3-28.

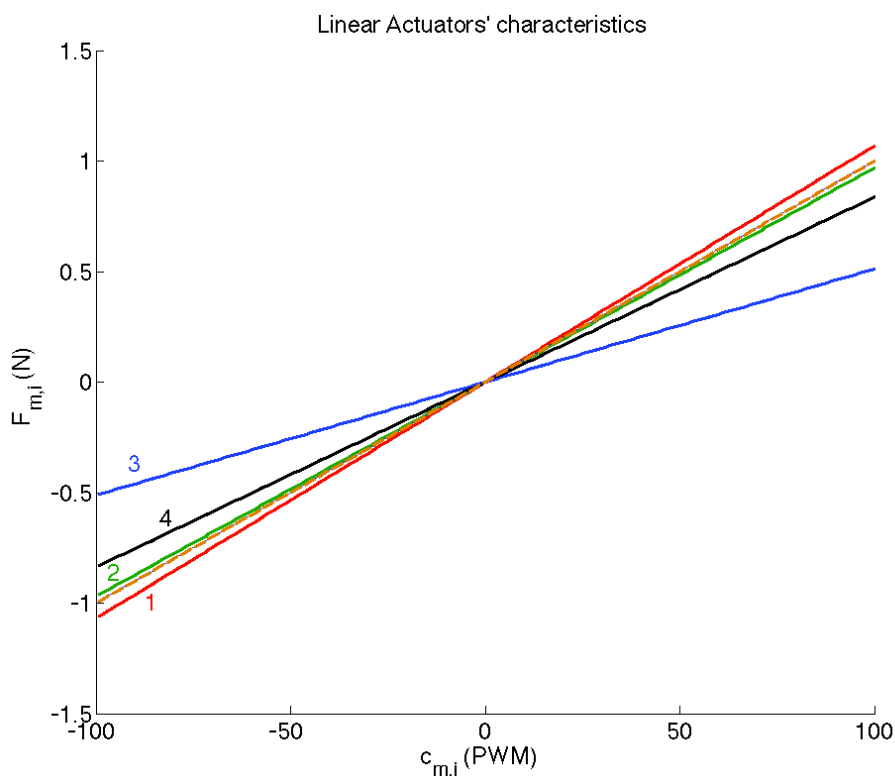


Figure 3-28 : Caractéristiques (inconnues) linéaires des actionneurs. En pointillés oranges la caractéristique

Dans ce cas linéaire, les matrices \mathcal{M}_{lin} et $\widehat{\mathcal{M}}_{lin}^{-1}$ s'écrivent :

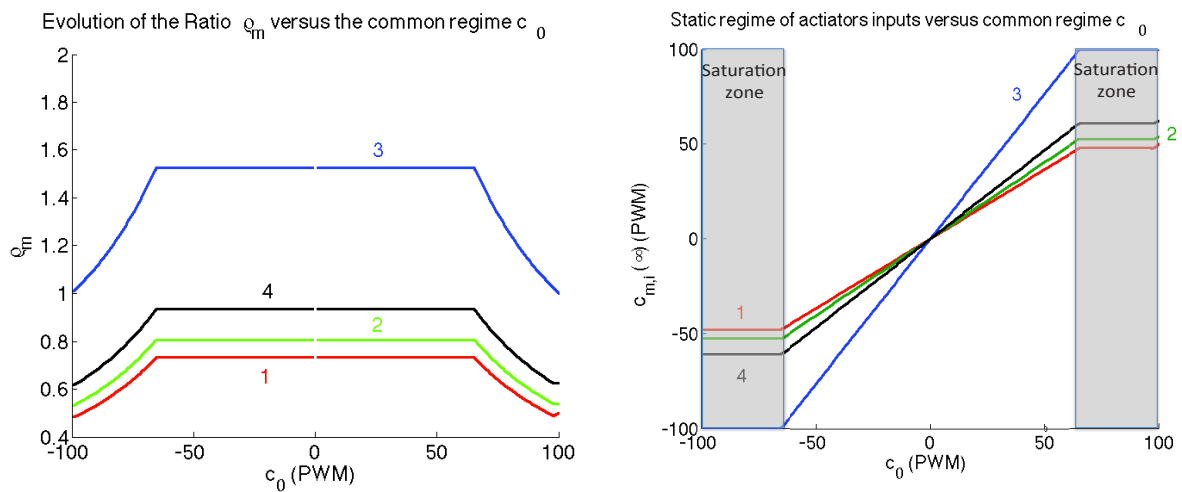
$$\mathcal{M}_{lin} = \begin{bmatrix} \eta_1 & 0 & 0 & 0 \\ 0 & \eta_2 & 0 & 0 \\ 0 & 0 & \eta_3 & 0 \\ 0 & 0 & 0 & \eta_4 \end{bmatrix}, \widehat{\mathcal{M}}_{lin}^{-1} = \begin{bmatrix} \eta_0^{-1} & 0 & 0 & 0 \\ 0 & \eta_0^{-1} & 0 & 0 \\ 0 & 0 & \eta_0^{-1} & 0 \\ 0 & 0 & 0 & \eta_0^{-1} \end{bmatrix} \quad (3.67)$$

Où d'après la figure 28, $\eta_1 = 1.0688 \cdot 10^{-2}$, $\eta_2 = 9.6940 \cdot 10^{-3}$, $\eta_3 = 5.1190 \cdot 10^{-3}$, $\eta_4 = 8.3710e \cdot 10^{-3}$ et $\eta_0 = 1 \cdot 10^{-2}$. Avec ces données, et dans le cadre de la simulation, on peut évaluer la relation (3.63). Il vient :

$$\mathbf{F}_B = \begin{bmatrix} 8.4680e - 01 & -9.7774e - 02 & 4.0633e - 01 \\ -3.2591e - 02 & 8.4680e - 01 & 3.8078e - 01 \\ 2.7731e - 02 & 7.7964e - 02 & 8.4680e - 01 \end{bmatrix} \cdot \mathbf{F}_B^d \quad (3.68)$$

L'équation (3.68) indique clairement les couplages occasionnés par la disparité des caractéristiques des actionneurs.

L'évaluation de la matrice \mathbb{Q} se fait au travers des expériences précédemment décrites et permet d'établir les coefficients de correction de la Figure 3-29-a



a) Valeur des coefficients de correction

b) Régimes statiques versus régime commun

Figure 3-29 : Evolution des coefficients de correction en fonction des régimes moteurs communs (a) ; Evolution des régimes statiques en fonction des régimes moteurs communs (b), cas linéaire.

On peut noter sur la Figure 3-29-a que l'évolution des coefficients de correction est constante dans la plage des régimes communs qui n'engendrent pas de saturation. On vérifie les plages de saturations sur la Figure 3-29-b où sont présentées les valeurs statiques atteintes par les actionneurs. Les coefficients de correction constants s'expliquent de par le fait qu'ils sont (linéairement) calculés à partir de situations où les moteurs produisent le même effort. Puisque les caractéristiques sont linéaires, le rapport des $c_{m,i}$ correspondants à un effort identique et constant, quel que soit cet effort et qu'aucun $c_{m,i}$ ne sature.

Ainsi, on en déduit que dans la plage de fonctionnement où les moteurs ne saturent pas :

$$\mathbb{Q}_{stat} = \begin{bmatrix} 0.7316 & 0 & 0 & 0 \\ 0 & 0.8067 & 0 & 0 \\ 0 & 0 & 1.5276 & 0 \\ 0 & 0 & 0 & 0.9341 \end{bmatrix}$$

au passage, on remarque que $\mathcal{M}_{lin} \cdot \mathbb{Q}_{stat} \cdot \widehat{\mathcal{M}}_{lin}^{-1} = 0.781981 \cdot \mathbb{I}_4$, ce qui explique le découplage.

En effet, l'utilisation des coefficients de correction revient à intercaler la matrice $\mathbb{Q}(\mathbf{c}_{m,0})$ comme décrit en (3.69)

$$\mathbf{F}_B = \mathbb{C} \cdot \mathcal{M} \cdot \mathbb{Q} \cdot \widehat{\mathcal{M}}^{-1} \cdot \mathbb{R} \cdot \begin{bmatrix} \mathbf{F}_B^d \\ r_m \end{bmatrix} \quad (3.69)$$

La linéarité des caractéristiques explique que la matrice \mathbb{Q} soit constante, dans les plages de fonctionnement qui n'engendrent pas de saturation, comme l'indique clairement la Figure 3-29-b, où la valeur des coefficients de correction se retrouve sur la pente des droites, pour chaque moteur. Or, pour chaque essai, s'il ne sature pas on peut dire que, pour un régime moteur donné c_0 :

$$F_1 = \eta_1 \cdot c_{m,1}^\infty = F_2 = \eta_2 \cdot c_{m,2}^\infty = F_3 = \eta_3 \cdot c_{m,3}^\infty = F_4 = \eta_4 \cdot c_{m,4}^\infty \quad (3.70)$$

puisque le système se stabilise avec des vitesses nulles, et donc que les moteurs exercent nécessairement la même poussée, et ce pour tout c_0 . Divisant les termes de la relation (3.70) par c_0 , il vient :

$$\eta_1 \cdot \rho_{m,1} = \eta_2 \cdot \rho_{m,2} = \eta_3 \cdot \rho_{m,3} = \eta_4 \cdot \rho_{m,4} \quad (3.71)$$

Or le terme $\mathcal{M} \cdot \mathbb{Q} \cdot \widehat{\mathcal{M}}^{-1}$ de la relation (3.69) peut s'écrire :

$$\mathcal{M} \cdot \mathbb{Q} \cdot \widehat{\mathcal{M}}^{-1} = \begin{bmatrix} \frac{\eta_1}{\eta_0} \cdot \rho_{m,1} & 0 & 0 & 0 \\ 0 & \frac{\eta_2}{\eta_0} \cdot \rho_{m,2} & 0 & 0 \\ 0 & 0 & \frac{\eta_3}{\eta_0} \cdot \rho_{m,3} & 0 \\ 0 & 0 & 0 & \frac{\eta_4}{\eta_0} \cdot \rho_{m,4} \end{bmatrix} \quad (3.72)$$

où, d'après (3.71), les termes diagonaux sont égaux. Ainsi, il vient :

$$\mathcal{M} \cdot \mathbb{Q} \cdot \widehat{\mathcal{M}}^{-1} = \frac{\eta_1}{\eta_0} \cdot \rho_{m,1} \cdot \mathbb{I}_4 = k_{\mathbb{Q}} \cdot \mathbb{I}_4 \quad (3.73)$$

Ce qui explique le phénomène de découplage, dans le cas linéaire. Les effets de découplage dans le cas non linéaire sont dus à cette approximation linéaire. Il faut cependant noter que la consigne \mathbf{F}_B^d n'est pas parfaitement appliquée puisque, considérant (3.73) et (3.69) il vient, comme tâche primaire :

$$\mathbf{F}_B = k_{\mathbb{Q}} \cdot \mathbf{F}_B^d$$

Le scalaire k_Q peut être considéré comme le ‘gain statique’ que la méthode induit. Par contre, le découplage des DDL est assuré.

3.5.5.4 Expérimentations



Figure 3-30 : Régulation des vitesses u et v de façon visuelle

Dans un premier temps, nous avons réalisé des expérimentations sans coefficient de correction, ni asservissement du cap afin d’observer la disparité des caractéristiques moteur, en nous limitant à la partie positive des caractéristiques. La Figure 3-31 permet de visualiser cette disparité. Ne disposant pas de capteur pour la mesure des vitesses linéaires seules les données relatives à la mesure du cap (ψ) et de sa vitesse (r) par la centrale d’attitude sont exploitées pour observer les mouvement du robot, où la courbe violette représente la vitesse moyenne de rotation autour de l’axe z_B .

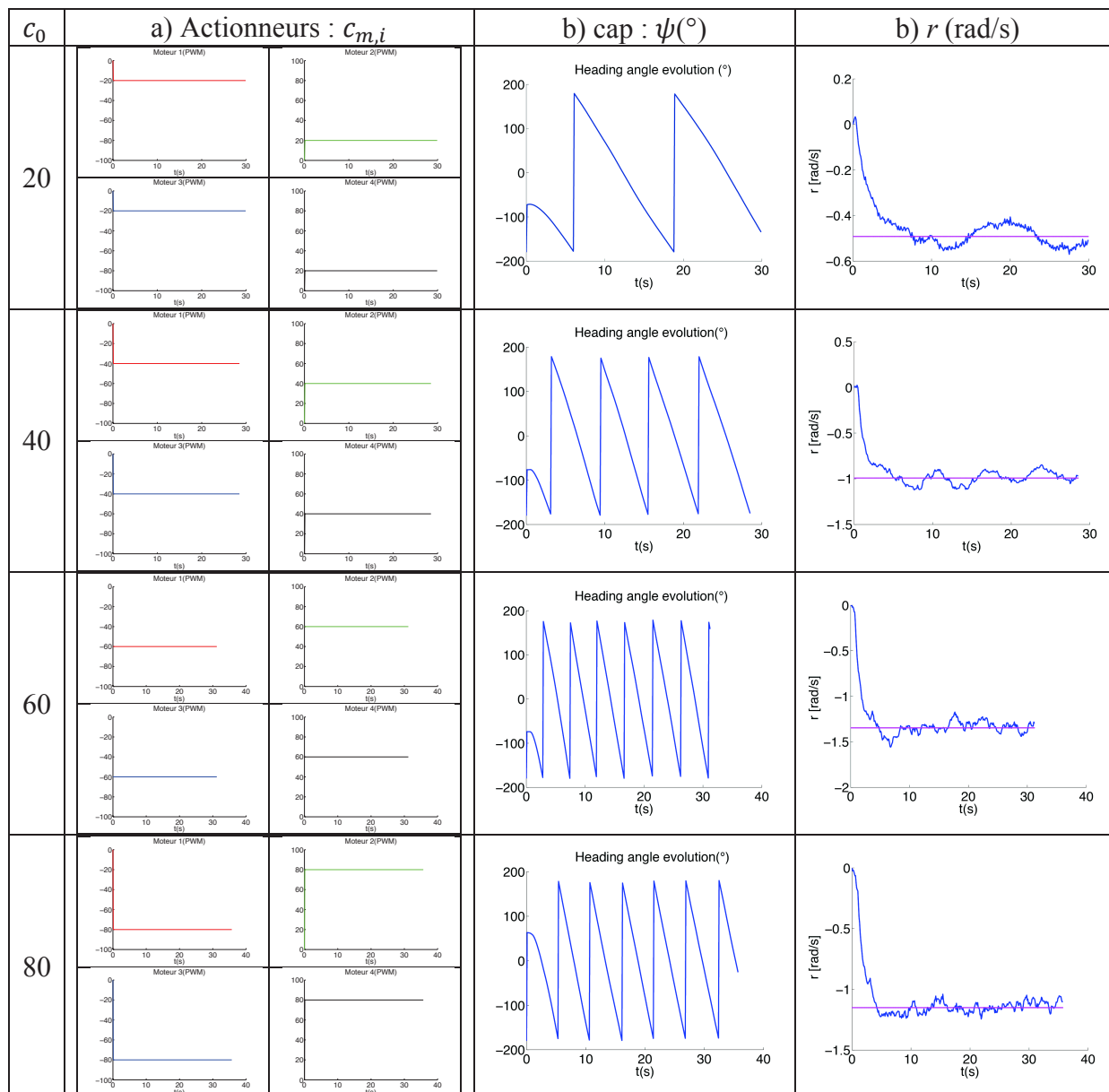


Figure 3-31 : Mise en évidence des effets de la disparité des caractéristiques moteurs sur le jack version 6 moteurs

Nous pouvons constater que le robot réalise des rotations sur lui même lorsque l'on applique le même $c_{m,i} = c_0$ à tous les moteurs. Cette expérimentation a été réalisée pour différents $c_0 = 20, 40, 60, 80$ PWM. Nous constatons sur la Figure 3-32 illustrant la vitesse moyenne pour chacun des c_0 , que plus le c_0 augmente plus la vitesse de rotation devient importante, cette évolution de la vitesse de rotation est relativement linéaire jusqu'à $c_0 = 60$ PWM.

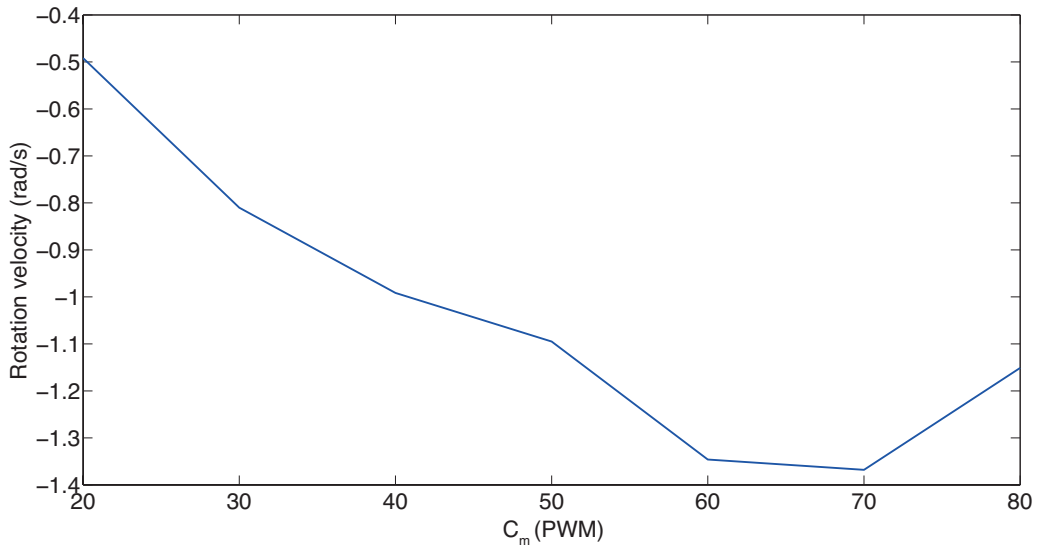


Figure 3-32 : Vitesse de rotation (rad/s) du robot pour des c_0 identiques à chacun des moteurs

Comme vu précédemment à la Figure 3-26, la Figure 3-33 nous montre l'effet de couplage lié à la disparité des caractéristiques des actionneurs. La Figure 3-33 illustre ce couplage de façon expérimentale avec $\mathbf{F}_B^d = [6,0,0]^T$ et $\mathbf{F}_B^d = [0,6,0]^T$.

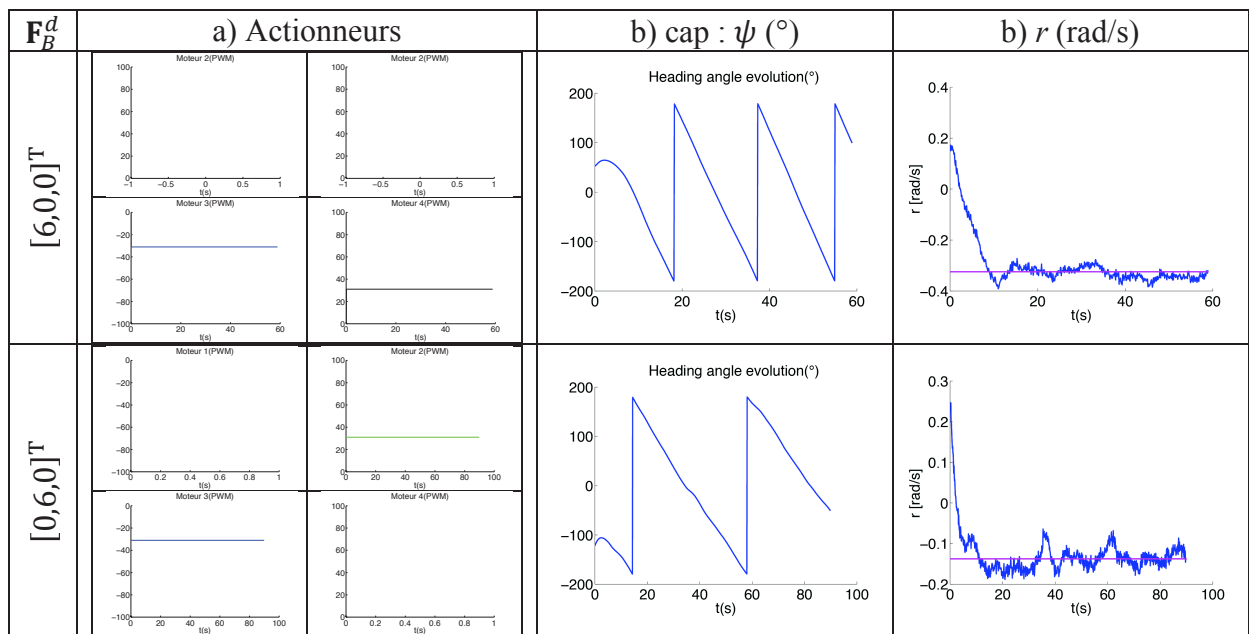


Figure 3-33 : Effet de couplage de la disparité des caractéristiques des actionneurs (expérimentation)

Dans la Figure 3-17, nous avons constaté que lorsque l'on applique un asservissement du cap, la disparité des caractéristiques entraînant la rotation était supprimée, mais on observait quand même des translations du système. La Figure 3-34 illustre de façon expérimentale les résultats obtenus à la Figure 3-17.

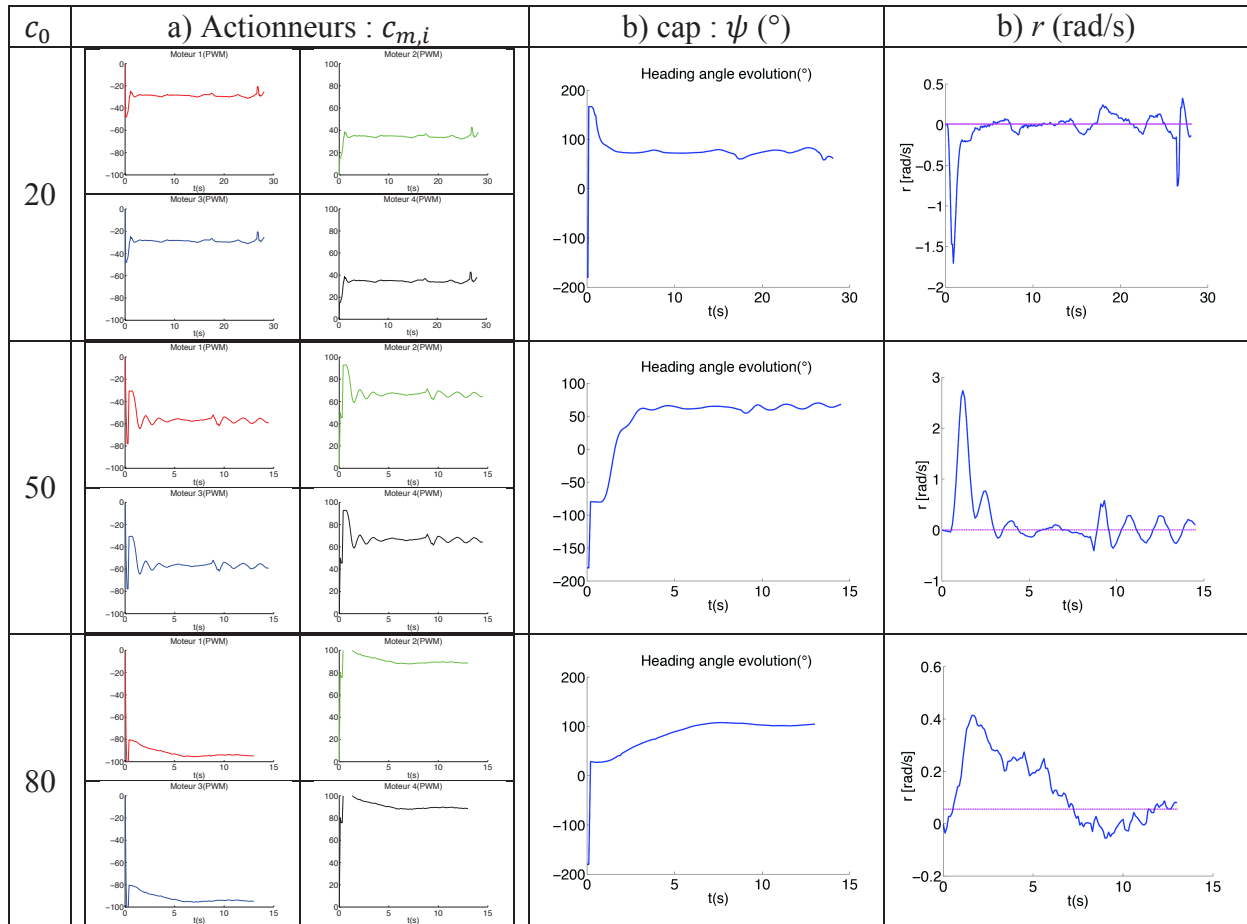


Figure 3-34 : Résultats avec asservissement du cap (expérimentation)

Nous pouvons constater que la rotation a bien été supprimée par l'asservissement dans la Figure 3-34, mais du fait de l'absence de mesure de la position du système ou du moins ses vitesses linéaires, nous pouvons simplement constater d'une façon visuelle la présence de vitesses linéaires parasites.

Pour supprimer les vitesses linéaires parasites, nous réalisons le protocole d'identification des coefficients de correction tel que nous l'avons précédemment décrit. Cependant, le Jack n'a pas la capacité d'emport qui permettrait l'utilisation d'un loch doppler pour réaliser l'asservissement des vitesses linéaires. Ainsi, nous utilisons les mesures de la centrale inertielle pour l'asservissement en cap, et reproduisons au joystick les effets de la régulation en vitesse, visuellement.

Nous avons réalisé ces tests pour différents régimes moteurs : $r_m = 2, 4, 6, 8$ et 10 N et nous avons reporté les résultats dans la Figure 3-35. Dans cette Figure 3-35, nous pouvons observer que la vitesse de rotation (r) oscille autour de zéro.

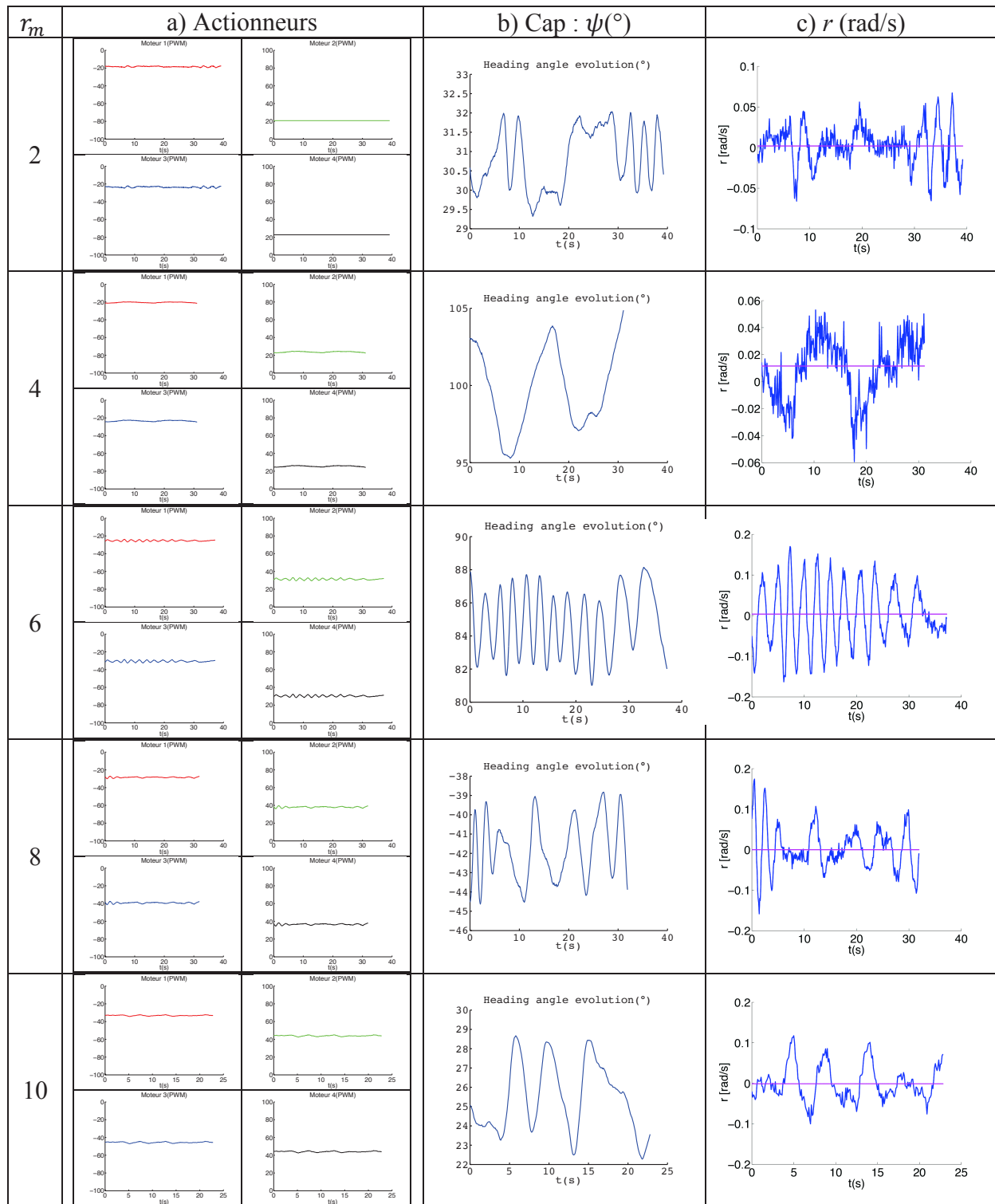


Figure 3-35 : Résultats expérimentaux de l'asservissement autour de régimes moteurs différents.

Dans la Figure 3-36, nous pouvons observer que la vitesse moyenne de rotation du système établie grâce à la courbe violette de la Figure 3-35 (b) (valeur moyenne de la vitesse de rotation autour de l'axe z_B) se stabilise autour d'une vitesse nulle.

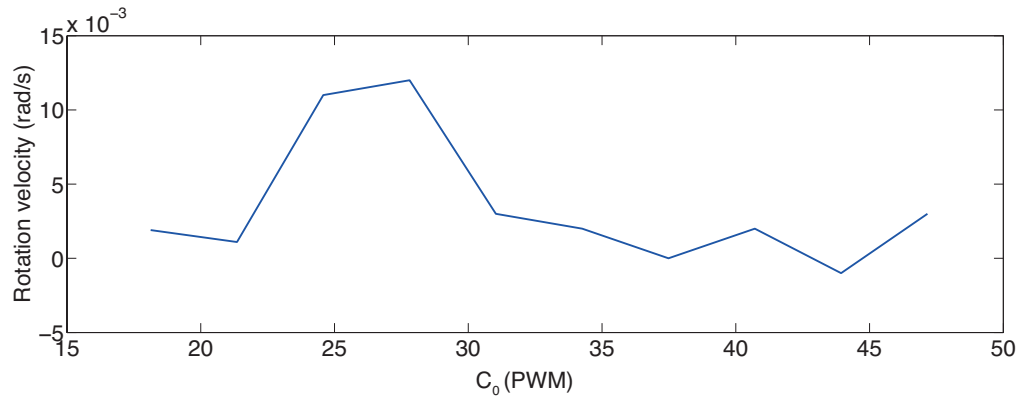


Figure 3-36 : Variation de la vitesse de rotation par rapport au régime commun.

En analysant les résultats de la Figure 3-35, nous pouvons identifier les différentes valeurs statiques atteintes par les entrées des moteurs ($c_{m,i}^\infty$) comme reporté dans le Tableau 3-9.

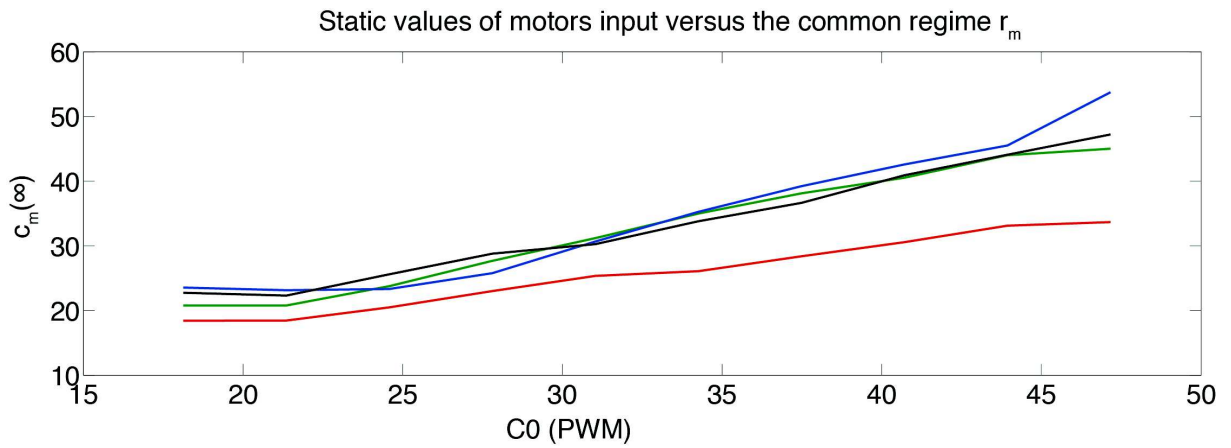


Figure 3-37 : Régime statique atteint par les moteurs, pour différents régimes commun.

La Figure 3-37 permet de voir l'évolution des régimes statiques atteints par les moteurs.

r_m		2	3	4	5	6	7	8	9	10	11
$c_{m,i}^\infty$	1	18.43	18.46	20.49	23.02	25.36	26.08	28.39	30.57	33.12	33.67
	2	20.79	20.79	23.77	27.69	31.20	35.00	38.12	40.54	44.01	45.02
	3	23.55	23.15	23.34	25.78	30.68	35.26	39.22	42.59	45.52	53.75
	4	22.75	22.31	25.60	28.79	30.26	33.81	36.64	40.92	44.09	47.22
c_0	0	18.13	21.35	24.58	27.80	31.03	34.26	37.48	40.71	43.94	47.16

Tableau 3-9 : Régime statique atteint par les asservissements en expérimentation

Nous obtenons les coefficients de correction de la Figure 3-38 en utilisant l'équation (46), et reportons les résultats dans le Tableau 3-10.

r_m	2	3	4	5	6	7	8	9	10	11	
$\rho_{m,i}$	1	1.0166	0.8644	0.8336	0.8279	0.8172	0.7613	0.7574	0.7509	0.7538	0.7139
	2	1.1468	0.9735	0.9670	0.9958	1.0054	1.0217	1.0170	0.9958	1.0017	0.9546
	3	1.2990	1.0841	0.9495	0.9271	0.9886	1.0292	1.0463	1.0462	1.0361	1.1397
	4	1.2549	1.0447	1.0415	1.0354	0.9751	0.9869	0.9775	1.0052	1.0035	1.0012
c_0	0	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000

Tableau 3-10 : Les coefficients de correction pour un régime commun r_m

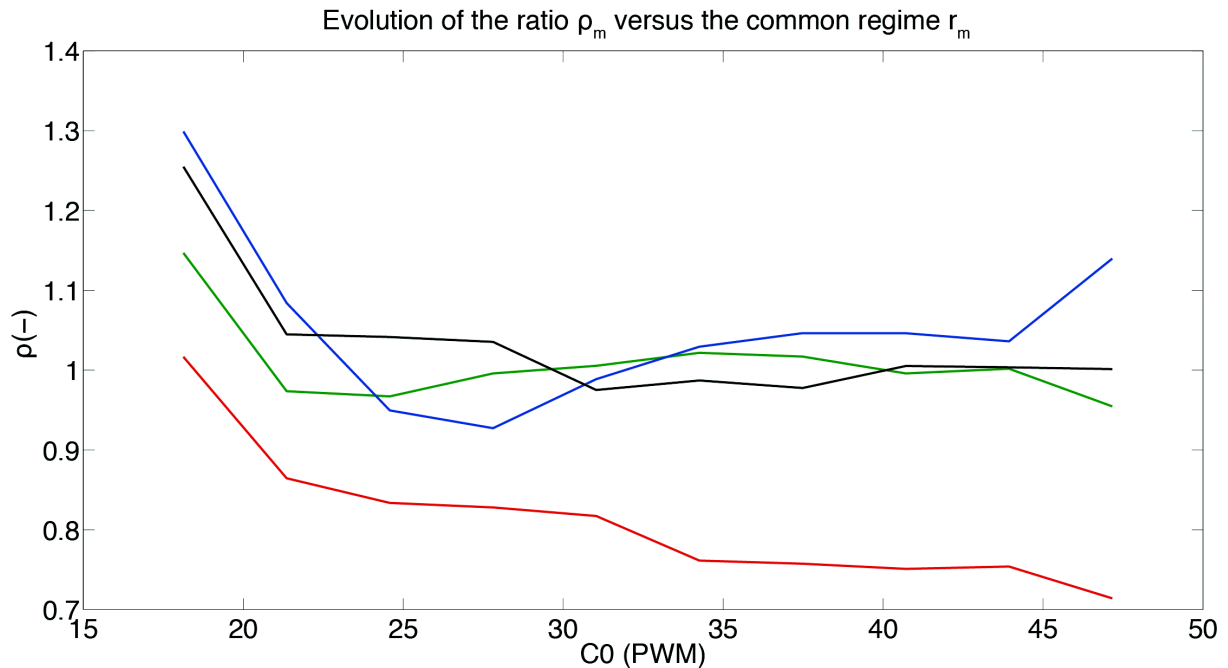


Figure 3-38 : Coefficients de correction expérimentalement établis sur le Jack 6.

Nous constatons à la Figure 3-38, que les caractéristiques ne sont pas identiques entre les différents moteurs et que le moteur 1 est plus puissant que les autres.

Dans la Figure 3-39, nous utilisons la matrice de correction \mathbb{Q} et nous appliquons sur le système en boucle ouverte la consigne de régime $c_0 = 24.58$ et 37.48 PWM. Nous pouvons constater que le robot ne bouge pas et que r est proche de zéro, de même à quelques oscillations près qui nécessiteraient un réglage plus fin des gains, le cap oscille autour de sa consigne.

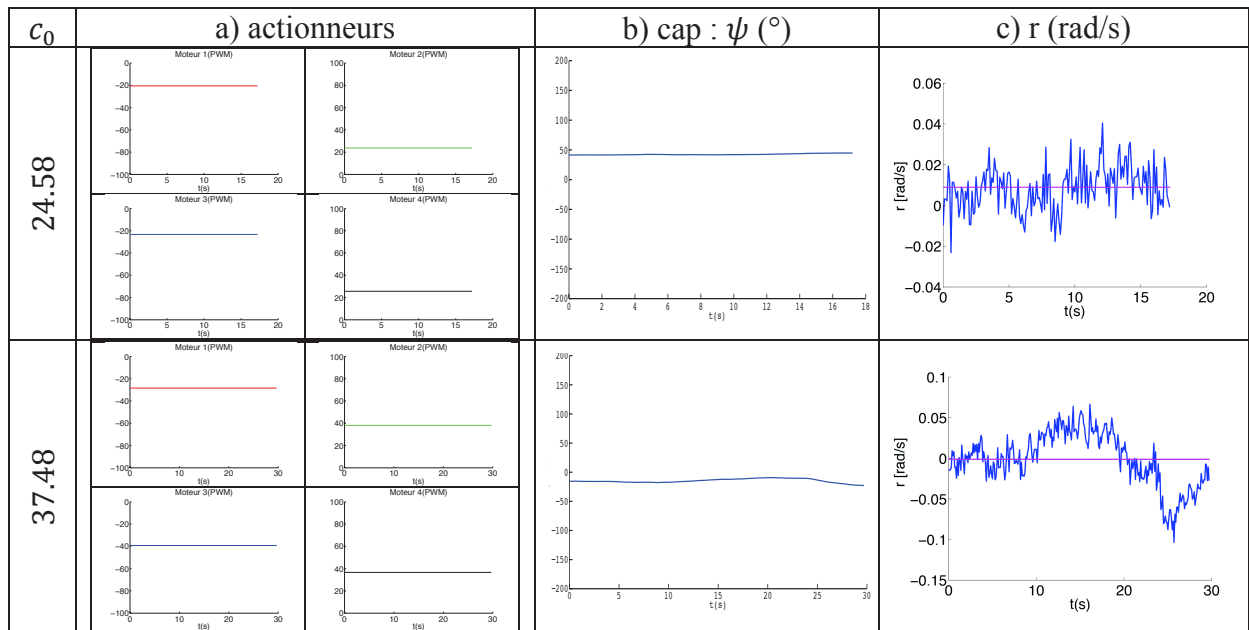


Figure 3-39 : Réponse du système à différents régimes communs, avec correction (expérimentation).

Enfin, nous appliquons ces coefficients comme décrit dans la section précédente, pour obtenir les résultats de la Figure 3-40, pour différentes valeurs désirées de F_B^d . Nous constatons un comportement du système découplé, comme attendu.

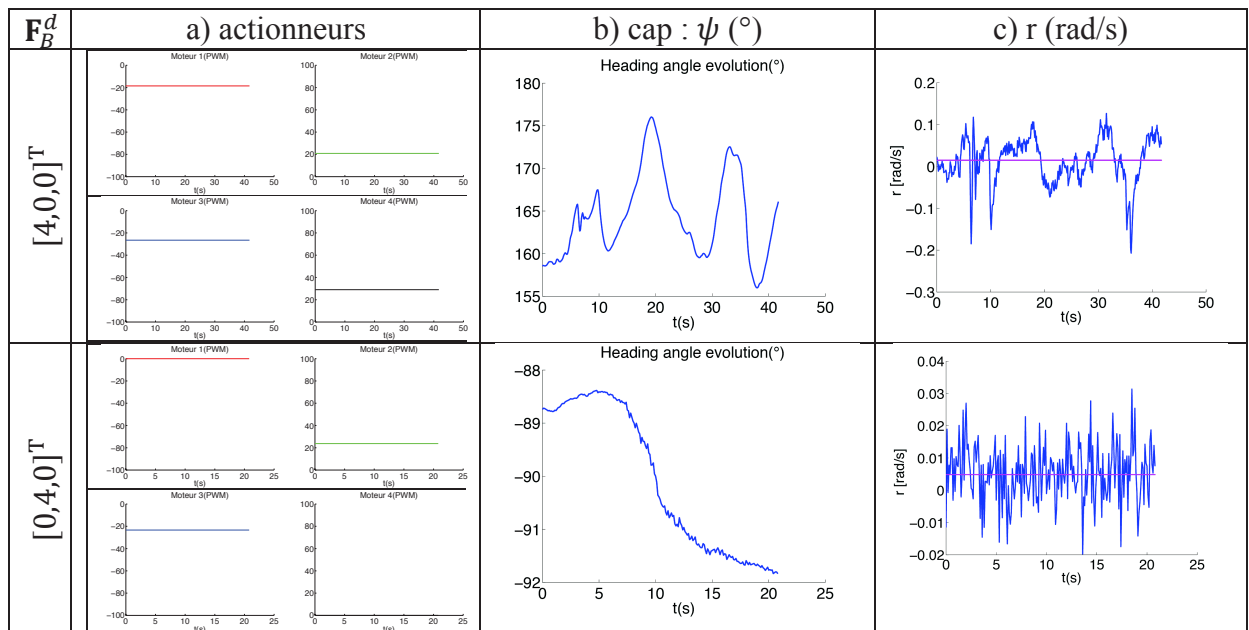


Figure 3-40 : Effet de découplage du système grâce au contrôle en boucle ouverte (expérimentation)

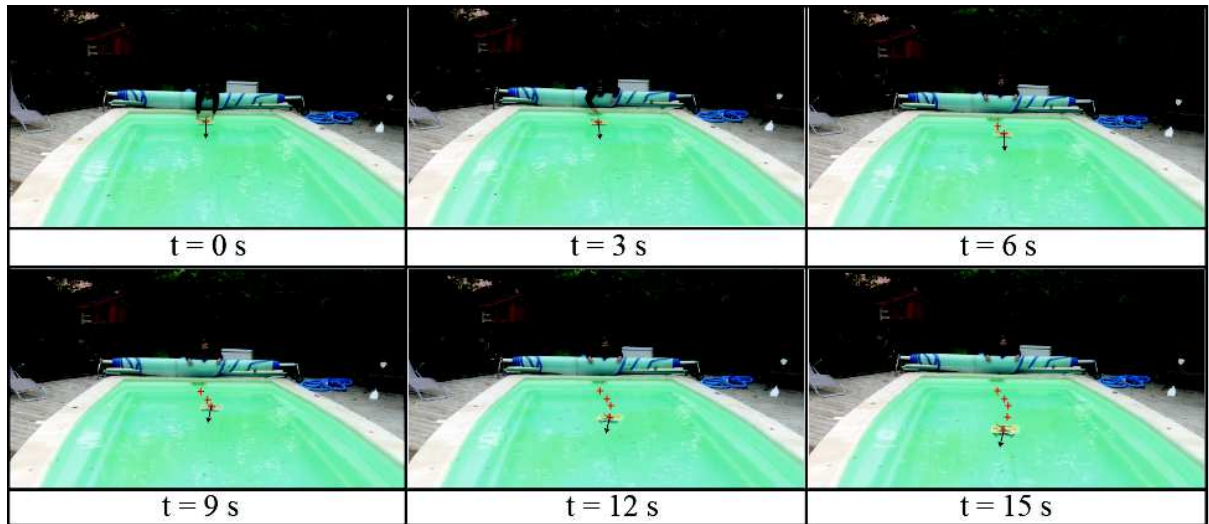


Figure 3-41 : Effet de découplage du système grâce au contrôle en boucle ouverte avec $\mathbf{F}_B^d = [4, 0, 0]^T$

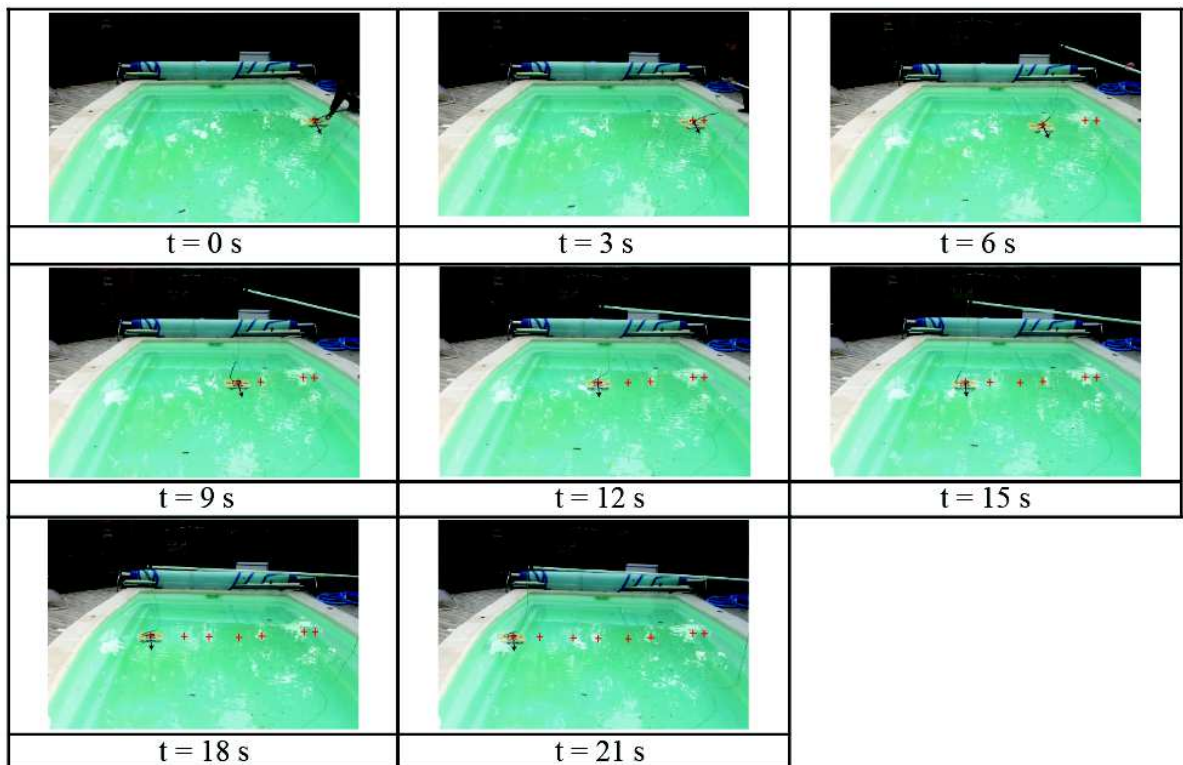


Figure 3-42 : Effet de découplage du système grâce au contrôle en boucle ouverte avec $\mathbf{F}_B^d = [0, 4, 0]^T$

Comme nous pouvons le constater dans la Figure 3-40, le robot a une bonne tenue du cap et de u et v ce que l'on peut également visualiser dans la Figure 3-41 et la Figure 3-42. Le bruit que l'on observe sur le cap dans la Figure 3-40 et Figure 3-41 est provoqué par le câble ombilical, bruit qui disparaît en partie lorsque l'on utilise une perche pour maintenir le câble

(Figure 3-42). Lorsque nous comparons cette Figure 3-40 avec la Figure 3-33, nous constatons la nette amélioration de la tenue de cap ainsi que des vitesses linéaires en boucle ouverte.

3.5.6 Généralisation au Jack 12

Nous allons maintenant nous intéresser au Jack équipé de son extension le faisant ainsi passer maintenant de 6 à 12 moteurs, il est représenté à la Figure 3-43.

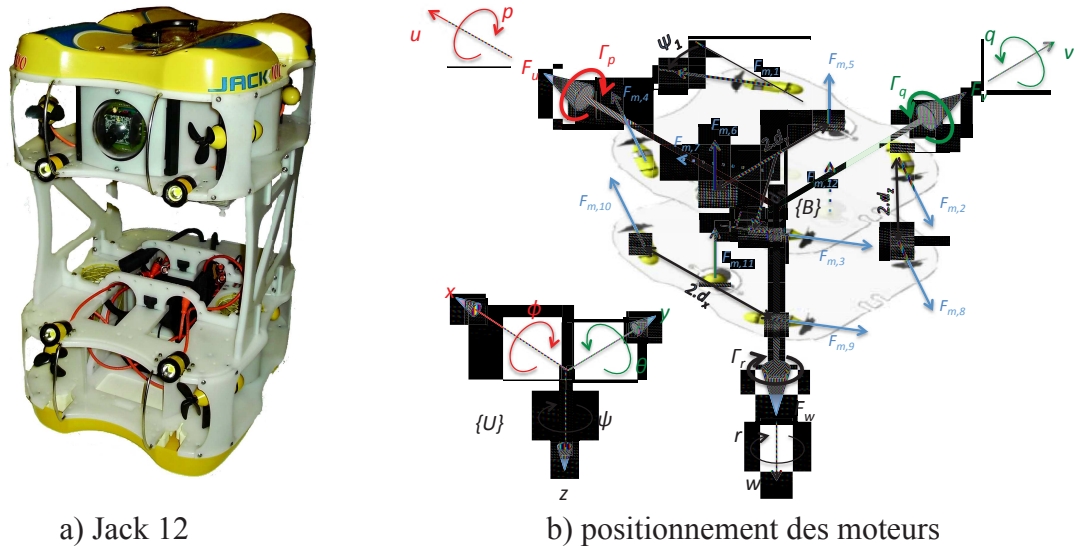


Figure 3-43 : le Jack 12.

Ce système présente une redondance d'actionnement que nous allons étudier. Dans un premier temps nous allons nous intéresser au Jack 12, sans considérer les propulseurs verticaux. Nous doublons donc l'étage d'actionnement précédemment étudié. Dans ce cas, \mathbf{F}_B est de dimension 5, \mathbf{F}_p et \mathbf{c}_m sont de dimension 8 :

$$\begin{aligned} \mathbf{F}_B &= [F_u, F_v, F_p, F_q, F_r]^T \\ \mathbf{F}_p &= [F_1, F_2, F_3, F_4, F_7, F_8, F_9, F_{10}]^T \\ \mathbf{c}_m &= [c_1, c_2, c_3, c_4, c_7, c_8, c_9, c_{10}]^T \end{aligned} \quad (3.74)$$

Le concentrateur \mathbb{C} est calculé comme à l'équation (3.19).

$$\mathbf{F}_B = \mathbb{C} \cdot \mathbf{F}_p \quad (3.75)$$

Avec, pour notre système :

$$\mathbb{C} = \begin{bmatrix} -0.866 & -0.866 & 0.866 & 0.866 & -0.866 & -0.866 & 0.866 & 0.866 \\ -0.5 & -0.5 & -0.5 & -0.5 & -0.5 & -0.5 & -0.5 & -0.5 \\ 0.1 & -0.1 & -0.1 & 0.1 & -0.1 & 0.1 & 0.1 & -0.1 \\ 0.173 & 0.173 & -0.173 & -0.173 & -0.173 & -0.173 & 0.173 & 0.173 \\ 0.226 & -0.226 & 0.226 & -0.226 & 0.226 & -0.226 & 0.226 & -0.226 \end{bmatrix} \quad (3.76)$$

Le répartiteur géométrique \mathbb{R}_G est calculé comme la pseudo-inverse de \mathbb{C} . L'étude de la redondance offerte par le Jack 12 se fait par l'étude des propriétés du noyau de \mathbb{C} , comme nous l'avons vu précédemment. Ainsi, on peut dire que si $\mathbf{M}_m = \text{Ker}(\mathbb{C})$ comme à l'équation (3.40), alors :

$$\mathbf{F}_p = [\mathbb{R}_G \quad \mathbf{M}_m] \cdot \begin{bmatrix} \mathbf{F}_B^d \\ \mathbf{R}_m \end{bmatrix} \quad (3.77)$$

avec \mathbf{R}_m un vecteur quelconque de dimension $(m - n)$. Dans le cas du système composé des huit actionneurs horizontaux du Jack 12, \mathbf{R}_m est de dimension 3. Appliquant le contrôle (3.77) on obtient :

$$\mathbf{F}_B = \mathbb{C} \cdot \mathbb{R}_G \cdot \mathbf{F}_B^d \quad (3.78)$$

où $\mathbb{C} \cdot \mathbb{R}_G = \mathbb{I}_5$ si \mathbb{C} est *full row rank*. La redondance s'exploite par le choix du vecteur \mathbf{R}_m , dans notre cas de dimension 3 :

$$\mathbf{R}_m = [r_0 \quad r_1 \quad r_2]^T \quad (3.79)$$

Comme candidat à la matrice \mathbf{M}_m , nous proposons :

$$\mathbf{M}_m = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & -1 \\ 1 & 1 & 1 \\ 1 & 1 & -1 \\ 1 & -1 & -1 \\ 1 & -1 & 1 \\ 1 & -1 & -1 \\ 1 & -1 & 1 \end{bmatrix} \quad (3.80)$$

La vérification que $\mathbf{M}_m = \text{Ker}(\mathbb{C})$ est directe.

La structure de la matrice \mathbf{M}_m est intéressante :

- La première colonne représente un régime commun comme nous l'avons traité précédemment. Ainsi r_0 s'applique à tous les moteurs.
- La deuxième colonne exprime la redondance entre l'étage d'actionnement haut et l'étage d'actionnement bas. En effet, si les résultantes des efforts en haut et en bas sont toutes deux nulles, le système reste immobile. Ainsi, r_1 exprime la différence entre les régimes communs haut et bas.
- La troisième colonne représente la symétrie entre les moteurs 'pairs' et 'impairs', inversement appliquée en haut ou en bas. Ainsi, r_2 est la différence entre les moteurs pairs et impairs.

Ceci permet des applications intéressantes.

3.5.6.1 Tolérance aux fautes

Comme nous l'avons effectué dans la section consacrée à l'identification simultanée des caractéristiques des moteurs du Jack 6 (Figure 3-23), où nous avons imposé une consigne désirée au moteur 1 pour l'asservissement du système autour de l'origine. Pour le Jack 12, la redondance d'ordre 3 pourrait nous permettre d'imposer un régime différent à 3 moteurs. Ces régimes pouvant être nuls, le système pourrait ainsi stopper jusqu'à 3 moteurs défectueux et continuer à réaliser la tâche principale (3.78). Ainsi, comme à l'équation (3.60), nous pouvons imposer un régime nul aux trois moteurs i, j et k en imposant un vecteur \mathbf{R}_m adéquat. Nous construisons la matrice \mathbf{M}_0 à partir de \mathbf{M}_m (équation 3.80) :

$$\mathbf{M}_0 = \begin{bmatrix} \mathbf{M}_m(i, :) \\ \mathbf{M}_m(j, :) \\ \mathbf{M}_m(k, :) \end{bmatrix} \quad (3.81)$$

où $\mathbf{M}_m(i, :)$ représente la i^{me} ligne de la matrice \mathbf{M}_m . On construit aussi le vecteur \mathbf{F}_{annul} comme suit :

$$\mathbf{F}_{annul} = \begin{bmatrix} -\mathbb{R}_G(i, :) \\ -\mathbb{R}_G(j, :) \\ -\mathbb{R}_G(k, :) \end{bmatrix} \cdot \mathbf{F}_B^d \quad (3.82)$$

où $\mathbb{R}_G(i, :)$ représente la i^{me} ligne du répartiteur \mathbb{R}_G . Il est cependant prudent de vérifier le conditionnement de la matrice $\mathbb{C}_{/\{i,j,k\}} \cdot \mathbb{C}_{/\{i,j,k\}}^+$ construite d'après \mathbb{C} duquel ont été retirées les lignes i, j et k , et où $\mathbb{C}_{/\{i,j,k\}}^+$ exprime la pseudo-inverse de $\mathbb{C}_{/\{i,j,k\}}$. Le conditionnement de cette matrice, et la proximité de ses valeurs propres de 1 expriment la manoeuvrabilité du système sans ses moteurs i, j et k . Enfin, le vecteur de régime \mathbf{R}_m est calculé comme suit :

$$\mathbf{R}_m = \mathbf{M}_0^{-1} \cdot \mathbf{F}_{annul} \quad (3.83)$$

sous réserve que \mathbf{M}_0 soit inversible. Vu les symétries que présente le système, on peut dire que les situations suivantes représentent exhaustivement toutes les configurations de perte de 3 moteurs. Ainsi on peut distinguer les cas suivants :

- Les trois moteurs appartiennent au même étage, exemple générique 1, 2, 3. Dans ce cas, la matrice de manoeuvrabilité, \mathbf{A} peut s'écrire :

$$\mathbf{A}_{/\{1,2,3\}} = \mathbb{C}_{/\{1,2,3\}} \cdot \mathbb{C}_{/\{1,2,3\}}^+ \quad (3.84)$$

On vérifie que le déterminant de \mathbf{A} est nul. En effet, une de ses valeurs propre est nulle et couple certains des DDL. Ainsi, on peut dire que si le système perd 3 moteurs du même étage d'actionnement, il ne sera plus entièrement contrôlable.

- Deux des moteurs défectueux appartiennent à l'un des étages d'actionnement, le troisième à l'autre. Compte tenu des symétries du système, les exemples génériques considérés sont reportés dans le Tableau 3-11. Ainsi, d'après le Tableau 3-11, on peut effectivement perdre jusqu'à 3 moteurs si les deux du même étage d'actionnement ne sont pas sur la diagonale. Le troisième peut être n'importe lequel de l'autre étage d'actionnement.
- Cas de la perte de deux moteurs seulement : s'ils appartiennent tous les deux au même étage d'actionnement, il ne faut pas qu'ils soient diagonaux, comme l'indique le Tableau 3-12. S'ils appartiennent à des étages d'actionnement différents, il n'y a pas de contraintes.
- Cas de la perte d'un seul moteur, comme l'indique le Tableau 3-13, il n'y a pas de contraintes.

3 moteurs défectueux	1,2,8	1,2,9	1,3,8	1,3,9	1,4,7	1,4,9
$\det(\mathbb{R}_{G,\{i,j,k\}} \cdot \mathbb{R}_{G,\{i,j,k\}}^T)$	1	1	0	0	1	1
$\det(\mathbf{M}_0)$	-4	-4	0	0	-4	-4

Tableau 3-11 : Analyse de la manœuvrabilité en cas de perte de 3 moteurs

2 moteurs défectueux	1,2	1,3	1,7	1,8	1,9	1,10
$\det(\mathbb{R}_{G,\{i,j\}} \cdot \mathbb{R}_{G,\{i,j\}}^T)$	1	0	1	1	1	1

Tableau 3-12 : Analyse de la manœuvrabilité en cas de perte de 2 moteurs

1 moteur défectueux	1	2	3	4
$\det(\mathbb{R}_{G,\{i\}} \cdot \mathbb{R}_{G,\{i\}}^T)$	1	1	1	1

Tableau 3-13 : Analyse de la manœuvrabilité en cas de perte d'un moteur

Nous illustrons ceci dans les simulations de la Figure 3-44. Pour cette simulation, nous considérons avoir une parfaite connaissance des moteurs, i.e. $\mathcal{M} = \widehat{\mathcal{M}}$, et des caractéristiques sans zone morte. Nous réalisons un asservissement d'attitude avec des vitesses linéaires nulles.

$$\mathbf{F}_B^d = \begin{bmatrix} F_u^d = K_{P,u} \cdot (u_d - u) + K_{I,u} \cdot \int_0^t (u_d - u) \cdot dt \\ F_v^d = K_{P,v} \cdot (v_d - v) + K_{I,v} \cdot \int_0^t (v_d - v) \cdot dt \\ \Gamma_p^d = K_{P,p} \cdot (\phi_d - \phi) - K_{D,p} \cdot \dot{\phi} \\ \Gamma_q^d = K_{P,q} \cdot (\theta_d - \theta) - K_{D,q} \cdot \dot{\theta} \\ \Gamma_r^d = K_{P,r} \cdot (\psi_d - \psi) - K_{D,r} \cdot \dot{\psi} \end{bmatrix} \quad (3.85)$$

Pendant les 30 premières secondes de la simulation, nous imposons une attitude désirée de $\boldsymbol{\eta}_{d,1} = \left[\frac{\pi}{4}, \frac{\pi}{5}, \frac{\pi}{6} \right]^T$, avec tous les moteurs actifs. Les 30 secondes suivantes asservissent le système autour de l'origine, $\boldsymbol{\eta}_{d,0} = [0,0,0]^T$, mais en coupant l'action des moteurs 1,2 et 8. Entre 60 et 90 secondes, la consigne est à nouveau $\boldsymbol{\eta}_{d,1}$, considérant les moteurs 1,2 et 9 comme défectueux. Enfin, entre 90 et 120 secondes, la consigne est l'origine $\boldsymbol{\eta}_{d,0}$, en coupant les moteurs 1,2 et 7. Ainsi, tous les cas réalisables précédemment décrits sont considérés.

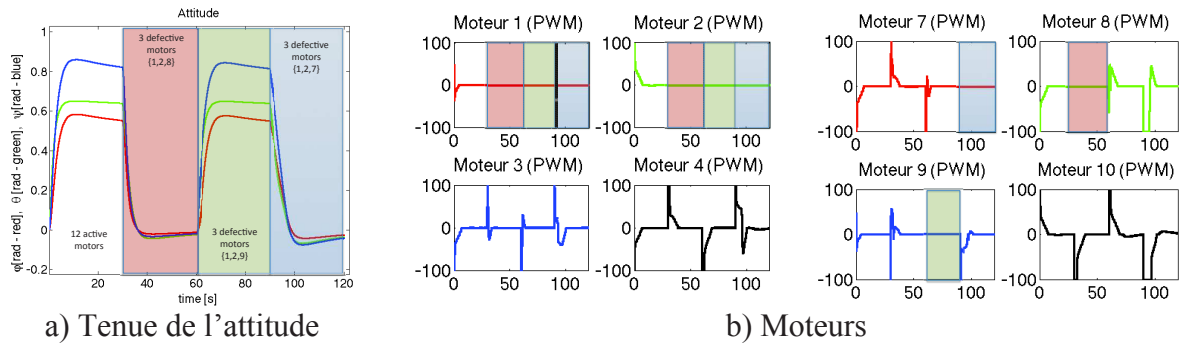


Figure 3-44 : comportement du système à l'extinction de 3 moteurs.

On constate une bonne tenue de la consigne d'attitude malgré l'extinction de 3 moteurs.

3.6 Points Clefs

- L'abstraction de l'étage d'actionnement apporte de la polyvalence à la plateforme. Cette abstraction est d'autant plus importante que le vecteur peut être configuré dans 2 versions : la version de base avec 6 moteurs et une version plus évoluée équipée de 12 moteurs. De fait, cette abstraction simplifie l'évolutivité du vecteur.
- L'abstraction d'actionnement est supportée par une entité appelée « répartiteur », qui a été validée en simulation dans un premier temps puis sur le vecteur dans le cadre d'expérimentations en piscine.
- La redondance dans l'étage d'actionnement est utilisée pour supprimer les zones mortes des propulseurs que nous avons identifiées lors de la caractérisation des propulseurs. Nous utilisons la redondance structurelle pour appliquer un régime moteur commun à l'ensemble des propulseurs sans que cela influe sur la commande désirée par l'opérateur.
- La disparité des caractéristiques des moteurs perturbe le fonctionnement attendu de l'étage d'actionnement vectoriel, qui devrait n'engendrer, en boucle ouverte, aucun mouvement à l'application d'un régime commun à tous les moteurs. Nous établissons un protocole d'estimation de coefficients de correction qui évite ce comportement parasite, et ce sans connaître aucune des caractéristiques des moteurs.
- Si toutefois la caractéristique d'un des moteurs est connue, alors le protocole expérimental précédent permet une identification partielle des caractéristiques des trois autres moteurs de l'étage d'actionnement.

- Nous montrons que l'exploitation de ces coefficients de correction permet d'assurer le découplage des DDL du système, dans le cas linéaire. Cette même exploitation appliquée au système réel montre les effets de ce découplage.
- Nous avons abordé l'extension du formalisme au cas du Jack 12 moteurs. Nous proposons une solution pour la tolérance aux fautes pour l'actionnement.

Chapitre 4 : Architecture de Contrôle

4.1 Introduction

L'architecture de contrôle doit également répondre à notre besoin de polyvalence, à ce titre elle doit permettre de modifier les entités logicielles constitutives, d'en ajouter, d'en retirer, ou de pouvoir les composer différemment selon le besoin applicatif... des propriétés « classiques » en génie logiciel. Mais le besoin est aussi de pouvoir accéder de manières différentes aux « fonctions » de l'architecture (souvent appelées tâches robotiques), par exemple piloter le déplacement du vecteur à distance via un lien de téléopération ou le piloter en embarqué par le biais d'un contrôleur haut-niveau (cf. chapitre 2). Les différents modes de pilotage peuvent d'ailleurs être « combinés » au sens de fonctions sous contrôle automatique (embarqué) et d'autres sous contrôle direct de l'opérateur (situation appelée co-contrôle), comme par exemple un contrôle de profondeur automatique et un contrôle d'avancée par l'opérateur. Cela est certes favorable, voire nécessaire, à la polyvalence mais cette « souplesse » impose de gérer correctement les multiples possibilités et accès aux ressources impliquées, comme notamment les ressources d'actionnement. Nous reviendrons sur ces aspects, mais dans l'immédiat le propos est de souligner que notre approche doit considérer :

- les exigences classiques de modularité, de composabilité, d'évolutivité et de réutilisabilité.
- la possibilité d'agréger ces compositions pour définir des « fonctions », elles-mêmes composables si besoin,
- la réification et la gestion de ressources pour en assurer une exploitation cohérente,
- et enfin, l'exécution temps-réel de cette architecture.

De nombreuses contributions scientifiques à l'interface entre le génie logiciel et la robotique ont été proposées depuis une vingtaine, voire une trentaine, d'années : un aperçu est donné dans [74], évoquant la diversité d'approches telles que, notamment, celles basées agents et celles basées composants [75], [76]. Si les approches sont diverses, elles convergent pour la plupart vers un modèle architectural selon 2 ou 3 niveaux : le niveau décisionnel (lié à la mission, la planification), le niveau exécutif (lié à l'exécution des fonctions) et le niveau fonctionnel (lié à l'instrumentation), ces deux derniers étant parfois regroupés.

Parmi les approches existantes, nous avons retenu :

- une approche de création de « fonctions » basée sur des entités logicielles, aux interfaces bien définies, qui soient composables et un middleware temps-réel associé offrant l'ensemble des mécanismes requis pour les composer (flux de données et d'événements) et les contrôler (flux de contrôle d'exécution). Il s'agit de l'approche modulaire offerte par le middleware temps-réel ContrAct (cf. section 0).
- une structuration de l'architecture selon 2 niveaux (décisionnel et exécutif).
- une abstraction des entités partagées, réifiées par l'intermédiaire de ressources.
- une agrégation des fonctions ou de leur composition basée sur le concept de service.

Notre choix est en effet d'organiser l'architecture selon une approche orientée service (Service Oriented Architecture). La SOA est une architecture logicielle organisée sur un ensemble de services simples, qui est fortement guidée par les aspects métier. Elle fait partie des architectures dites « logiques » [77], [78]. La SOA suit le principe relativement simple de l'offre et de la demande de service. Un « client » (ou consommateur ou demandeur) sollicite

un service offert par un « fournisseur » (ou producteur) au regard de la fonction qu’il offre, sans avoir à connaître comment le service est réalisé : le service encapsule la « logique » (modèles) selon laquelle il est réalisé. L’architecture orientée service est donc basée sur la description des entités qui la constitue et de leurs relations : la mise en relation entre le consommateur et le producteur peut être assurée par une entité dite médiateur (souvent selon une approche « bus ») ou un gestionnaire de services. Cette structuration d’une architecture autour de la notion de services est reconnue pour offrir une réponse efficace au besoin d’évolutivité. Jusqu’ici, les architectures orientées services étaient largement utilisées pour les applications informatiques [79], [80]. L’utilisation de la SOA pour les systèmes embarqués est un thème de recherche plus récent. Les concepts de SOA ont également été appliqués pour les systèmes embarqués, avec recomposition de services, et les applications liées à la robotique [81], [82], certains offrant un framework permettant de développer des services robotiques composables et réutilisables [83].

A titre d’exemple, l’Université de l’Arizona a proposé une architecture orientée service pour la robotique (Figure 4-1) [84].

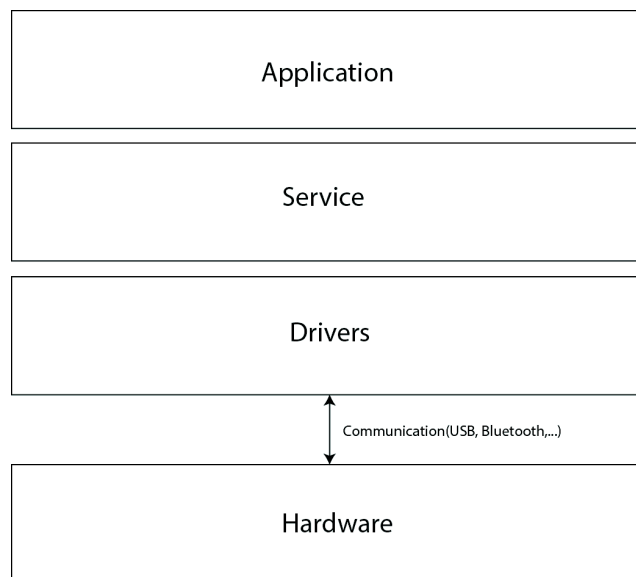


Figure 4-1 : Architecture Orienté Service appliquée à la robotique [84]

Cette architecture est découpée en 4 niveaux (Figure 4-1). La couche la plus basse représente la partie matérielle du robot, cela comprend les actionneurs et capteurs. Cette couche communique avec la couche driver qui est d’un niveau plus élevé via des liens de communication qui peuvent être divers (USB, WIFI, Bluetooth...). Ce niveau plus élevé contient l’ensemble des drivers qui permet de contrôler l’ensemble des périphériques sous-jacent contenu dans la couche matérielle. La troisième couche est la couche service qui constitue le cœur du système. La composition de l’ensemble des services est réalisée au niveau supérieur dans la couche application.

L’évolutivité est apportée par la vision organisationnelle de cette architecture, qui permet de :

- de faire évoluer les services (fonctions utilisées pour la réalisation du service offert) sans remettre en cause leurs compositions

- de construire des services de plus haut niveau (services agrégés) en combinant les différents services existants.
- et donc de favoriser naturellement la réutilisabilité des services à travers différentes compositions (i.e. services agrégés)

Ces éléments simples permettent de construire et d'organiser le système en le confrontant aux réalités métiers (composition dépendante de l'approche métier). La modularité et l'évolutivité de cette solution répondent, ou contribuent, au besoin de polyvalence de notre architecture à ce niveau d'abstraction.

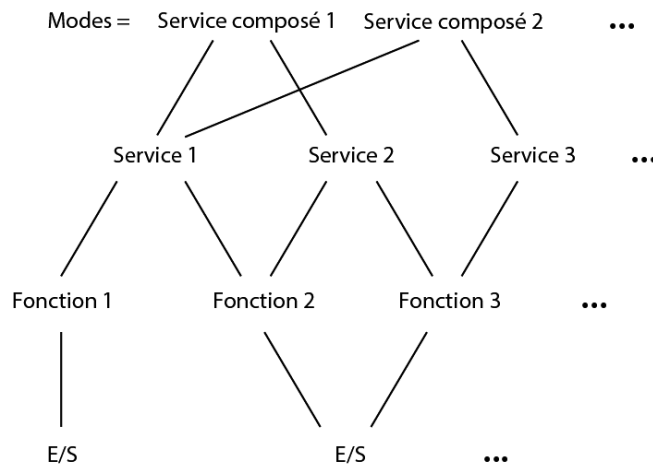


Figure 4-2 : Relation entre les services et les ressources

Néanmoins, comme nous l'avons évoqué, l'architecture doit également supporter la notion de ressource et fondamentalement permettre l'expression de contraintes entre services et ressources. En effet, si l'on prend comme exemple la représentation donnée Figure 4-2, il est difficile voir impossible d'assurer que les compositions de fonctions, puis de services, exploitent de manière cohérente les ressources disponibles (par exemple un actionneur). Il est donc nécessaire de réifier (donner existence) ces ressources afin de pouvoir exprimer et vérifier, des contraintes quant à l'utilisation des ressources à travers les compositions successives. Nous allons illustrer ces propos à travers les *modes de fonctionnement*.

4.2 Modes de fonctionnement et ressources

Au regard de ce que nous avons évoqué dans les chapitres précédents, l'architecture de contrôle doit prendre en compte différents modes de fonctionnement. Nous allons en évoquer quelques exemples, en nous attachant à mettre en exergue le lien avec les ressources :

- La télé-opération « bas-niveau directe » à travers laquelle l'opérateur contrôle directement les moteurs en transmettant directement les consignes moteurs. Cela requiert beaucoup de dextérité et il est difficile de contrôler le robot dans l'espace. Il est préférable et plus réaliste, au sens de la maniabilité, que ce contrôle soit exprimé en termes de degrés de liberté (DDL) du robot plutôt que de moteurs directement. C'est pour cela que nous avons mis en place l'étage d'actionnement et tout particulièrement le répartiteur (Chapitre 3). Le répartiteur est d'autant plus important qu'il doit assurer cette projection soit sur 6 soit sur 12 moteurs, selon que le robot est doté de l'extension d'actionnement ou non. Un tel contexte prohibe une téléopération directe des 12 moteurs. Le répartiteur permet

d'exploiter la redondance structurelle et ainsi piloter le système par plusieurs consignes selon les DDL du système impliqués dans le pilotage en cours.

- La télé-opération « bas-niveau indirecte » à travers laquelle l'opérateur contrôle le robot via des fonctions de contrôle embarquées telles que par exemple le suivi d'un cap. Dans ce cas l'opérateur ne fait que donner les consignes de cap et le contrôleur embarqué en assure le suivi. A ce niveau de télé-opération il est également possible pour l'opérateur de confier au contrôleur embarqué l'exécution de certaines fonctions de façon automatique, comme l'asservissement en profondeur par exemple. Dans ce cas l'opérateur donne la consigne de profondeur et le contrôleur embarqué en assure le suivi tout au long de l'exploration. Il apparaît donc que différents degrés de liberté du robot peuvent être simultanément sous des contrôles différents. Dans un souci de cohérence et de sécurité, nous introduirons donc le concept de *ressource DDL* ; il est en effet nécessaire de réifier et gérer ces ressources pour assurer qu'à tout instant un DDL n'est soumis qu'à un seul et unique contrôle. La réification des ressources DDL est d'autant plus nécessaire que, au delà des choix de l'opérateur de contrôler ou non certains DDL, certaines tâches, dont celles se référant à l'environnement, induisent un couplage entre les DDL (par exemple le suivi de fond).
- La télé-opération « haut-niveau » à travers laquelle l'opérateur pilote le robot à un niveau d'abstraction plus élevé. Dans ce mode les fonctions de contrôle sont plus avancées, comme par exemple le suivi d'une trajectoire définie. Ces fonctions avancées relèvent du contrôleur haut-niveau. Il est donc nécessaire de permettre au contrôleur haut-niveau de prendre le contrôle sur les DDL, et ce à travers le contrôleur bas-niveau qui est celui en relation effective avec les moteurs. De fait, nous devons permettre, au sein de l'architecture de contrôle bas-niveau, un mode « transparent » permettant cet accès direct.
- Le mode autonome dans le cadre duquel l'opérateur définit une mission que le robot doit accomplir de façon autonome. Cela suppose de définir un langage de spécification de mission, notamment en termes d'objectifs et de séquence d'objectifs à accomplir. Ce mode ne sera pas évoqué dans ce manuscrit, mais il est possible d'ores et déjà de préciser que différents types d'objectifs devront pouvoir être décrits. D'une part les objectifs *désirés*, i.e. ceux demandés par l'opérateur, qui sont de deux types : les objectifs que l'on peut qualifier de « classiques » et spécifiés dans l'univers du robot, comme par exemple atteindre un point dans l'espace (sous-marin), et les objectifs « métier » qui doivent pouvoir être spécifiés dans l'univers opérateur, i.e. en référence à des phénomènes perçus ou observés dans l'environnement (au sens par exemple de remonter un gradient de température selon un modèle). D'autre part les objectifs *obligés* qui proviennent de l'obligation pour le robot de répondre à une contrainte propre non spécifiée dans la mission comme, par exemple, éviter un obstacle, remonter en surface pour faire un recalage GPS de sa position (i.e. corriger sa position estimée, estimation qui dérive dans le temps). Ce mode autonome se rencontre aussi dans la situation de missions pour lesquelles la qualité du lien entre l'opérateur et le système est contextuellement variable (acoustique, rupture – volontaire ou pas – de l'ombilical).

De fait, les modes de fonctionnement que nous venons de décrire peuvent être assimilés à une composition de *services*.

Ces services correspondent en fait à l'activation de fonctions sur des ressources identifiées : ces fonctions correspondent à des relations entrées/sorties régissant le fonctionnement des ressources DDL qu'elles impliquent.

Ainsi l'opérateur peut solliciter des services offerts par le contrôleur bas-niveau, ou des services offerts par le contrôleur haut-niveau qui lui même fait potentiellement appel à des services du contrôleur bas-niveau. Selon cette conception, il devient possible de faire correspondre les évolutions du contrôle, et des interactions, aux évolutions du vecteur. La communication entre les services, et donc la connectique entre les caissons (contrôleur haut-niveau, contrôleur bas-niveau, capteurs, etc.), est un point important sur lequel nous reviendrons.

Donnons dans un premier temps les définitions des modes et des services, en s'attachant à mettre en exergue la relation avec les ressources, cette relation étant portée par les configurations. Tous ces termes vont être maintenant définis.

4.3 Définition d'un mode de fonctionnement et d'un service

Dans la suite du document, les notations suivantes sont utilisées :

- (\exists) symbolise la composition,
- $(\exists!)$ signifie 'il existe un unique'.

Ainsi, notre architecture est définie, à travers les *Modes*, par l'expression ci-dessous :

$$\begin{aligned}
 A \ni \{EM\}, EM \ni \{M_1, \dots, M_i, \dots, M_k\}, \exists! M_i \in \{M\} \\
 M_i \ni \{M_{id}, ES_j\} \\
 ES_j \ni \{S_{id_1}, \dots, S_{id_i}, \dots, S_{id_z}\}, \exists! S_{id_i} \in (\{S\} \& \{ES_j\})
 \end{aligned}
 \tag{4.1}$$

Avec :

- A l'architecture composée d'un ensemble de modes de fonctionnement défini par $\{M\}$
- M_i un mode, unique et appartenant à l'ensemble de modes $\{M\}$, décrit par :
 - o un identifiant M_{id} ,
 - o un ensemble de services défini par ES_i
- ES_i un ensemble de services composés ou non. Il est unique dans $\{S\}$. Chaque service identifié par S_{id_i} un service unique appartenant à l'ensemble des services défini par $\{S\}$.

Le *Service* est une entité logicielle essentielle de l'architecture qui, dans notre contexte, est définie comme une abstraction de la capacité du robot à réaliser certaines fonctions (actions) en mobilisant ou en créant un certain nombre de *Ressources*, selon les *Configurations*.

Fonctionnellement, au sens de l'architecture de contrôle, les services sont gérés par un unique gestionnaire de services.

Un service, dont le modèle est schématiquement représenté par la Figure 4-3, est défini par l'expression ci-dessous :

$$\begin{aligned}
S_j &\ni \{S_{id}, PS, EU_n, EC_j\}, \exists! S_j \in \{S\} \\
EU_n &\ni \{U_{id_1}, \dots, U_{id_i}, \dots, U_{id_k}\}, \exists! U_{id_i} \in (\{U\} \& \{EU_n\}) \\
EC_j &\ni \{C_{id_1}, \dots, C_{id_i}, \dots, C_{id_x}\}, \exists! C_{id_i} \in (\{C\} \& \{EC_j\})
\end{aligned}
\tag{4.2}$$

Avec :

- S_j un service, unique dans $\{S\}$.
- S_j un service composé (décrit par un ensemble de paramètres) :
 - un identifiant S_{id} ,
 - une priorité PS , prise en compte lorsqu'un conflit d'accès concurrent dans l'obtention des ressources se présente.
 - un ensemble des utilisateurs (Users) autorisés EU_n dont chaque User identifié par U_{id_i} appartient, de manière unique, à l'ensemble des utilisateurs défini par $\{U\}$. En effet, les différentes opérations disponibles sur un service telles que l'activation, le paramétrage et la désactivation sont potentiellement soumises à autorisation.
 - un ensemble de configurations EC_j , dont chaque configuration est identifiée par C_{id_i} appartient, de manière unique, à l'ensemble des configurations défini par $\{C\}$ (détaillé plus loin).

Les services sont composables, une composition de services ES_j (composée de services S_{id_i}) est définie dans l'expression (4.1).

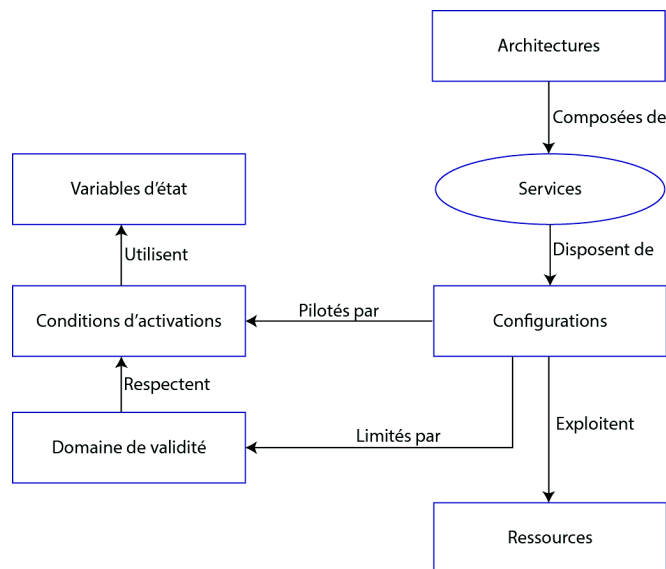


Figure 4-3 : Modèle du service

4.4 Définition de la configuration d'un service

Un service dispose de *Configurations* (Figure 4-3). Une configuration est une description de la manière de réaliser une fonction (action) avec la (les) ressource(s) associée(s).

Une configuration, schématiquement représentée par la Figure 4-3, est définie par l'expression suivante :

$$C_j \ni \{C_{id}, ER_k, UCA_j, D_j, PC\}$$

$$ER_k \ni \{R_{id_1}, \dots, R_{id_i}, \dots R_{id_y}\}, \exists! R_{id_y} \in \{R\}, ER_k \subseteq \{R\} \quad (4.3)$$

Avec :

- C_j la configuration décrite par (ses paramètres) :
 - o C_{id} l'identifiant de la configuration,
 - o ER_k le sous-ensemble des ressources mobilisées ou créées par le service dans la configuration désignée.
 - o UCA_j les conditions d'activation de la configuration C_j , déduites du domaine de validité D_j associé à la configuration C_j .
 - o PC le niveau de priorité qui permet de sélectionner une configuration lorsque l'on dispose de plusieurs solutions dans un contexte donné. La configuration la plus prioritaire sera alors utilisée.
- ER_k un sous-ensemble de ressources, chacune identifiée par R_{id_y} , unique dans $\{R\}$.

Cette formulation ne met pas explicitement en évidence le lien entre la configuration et la fonction (indifféremment appelée action, traitement, tâche robotique...). En fait cette relation est établie indirectement à travers les ressources associées. Nous l'expliciterons à travers la section 4.6, après avoir défini les ressources de manière générale.

Une configuration exploite/crée des *Ressources*, possède des *Conditions d'activation* et un *Domaine de validité*.

Les *Conditions d'activation* permettent de savoir si le service est activable ou non, selon le contexte. A titre d'exemple, le service de centrage ne peut être utilisé que lorsque l'on se trouve dans un environnement « fermé » (un conduit karstique ou une canalisation, par exemple), par conséquent cela n'a pas de sens de pouvoir l'activer ou de le laisser activé lorsque l'on se trouve dans un environnement « ouvert » puisqu'il n'est pas adéquat (vide karstique, navigation en pleine eau, suivi de parois, par exemple). Dans la mesure où un service peut avoir plusieurs configurations associées, les conditions d'activation d'un service sont l'union des conditions d'activation de ses configurations.

$$UCA \ni \{CA_1, \dots, CA_i, \dots CA_k\}, \exists! CA_i \in \{CA\}, UCA \subseteq \{CA\} \quad (4.4)$$

Avec :

- UCA une union de conditions d'activation, chacune identifiée par CA_i . UCA est un sous-ensemble de $\{CA\}$.
- CA_i , unique, appartenant à l'ensemble des conditions d'activation $\{CA\}$.

Le *Domaine de validité* d'une configuration définit les conditions d'activation, en se basant sur des variables d'état relatives soit au système 'restreint' à la fonction réalisée (par

exemple, fonctionnement « dégradé » ou « nominal ») soit au contexte (par exemple, environnement « fermé » ou « ouvert »).

Les *Conditions d'activation* forment donc un ensemble de contraintes à respecter pour l'activation, ou le maintien en activité, d'un service et de fait peuvent induire sa désactivation. Toutes les conditions d'activation CA_i contribuant à l'union des conditions d'activation UCA associée à la configuration C_j concernée, doivent être vérifiées (i.e., opérateur ET entre les CA_i).

L'évolution de la configuration d'un service peut donc être d'origine interne, à savoir un changement de configuration peut automatiquement être effectué au sein d'un service.

En effet, un service peut avoir plusieurs configurations possibles (donc plusieurs C_j au sein de EC_j) et les conditions de désactivation (respectivement d'activation) d'une configuration C_i au sein de EC_j peuvent correspondre aux conditions de d'activation (respectivement de désactivation) d'une configuration C_k au sein de EC_j . Par ce biais, i.e. par l'intermédiaire des conditions d'activation, il est possible de décrire la logique qui régit le changement de configuration au sein d'un service.

Fonctionnellement, au sens de l'architecture de contrôle, le changement de configuration au sein d'un service est géré par un gestionnaire de configuration ; chaque service a son gestionnaire de configuration.

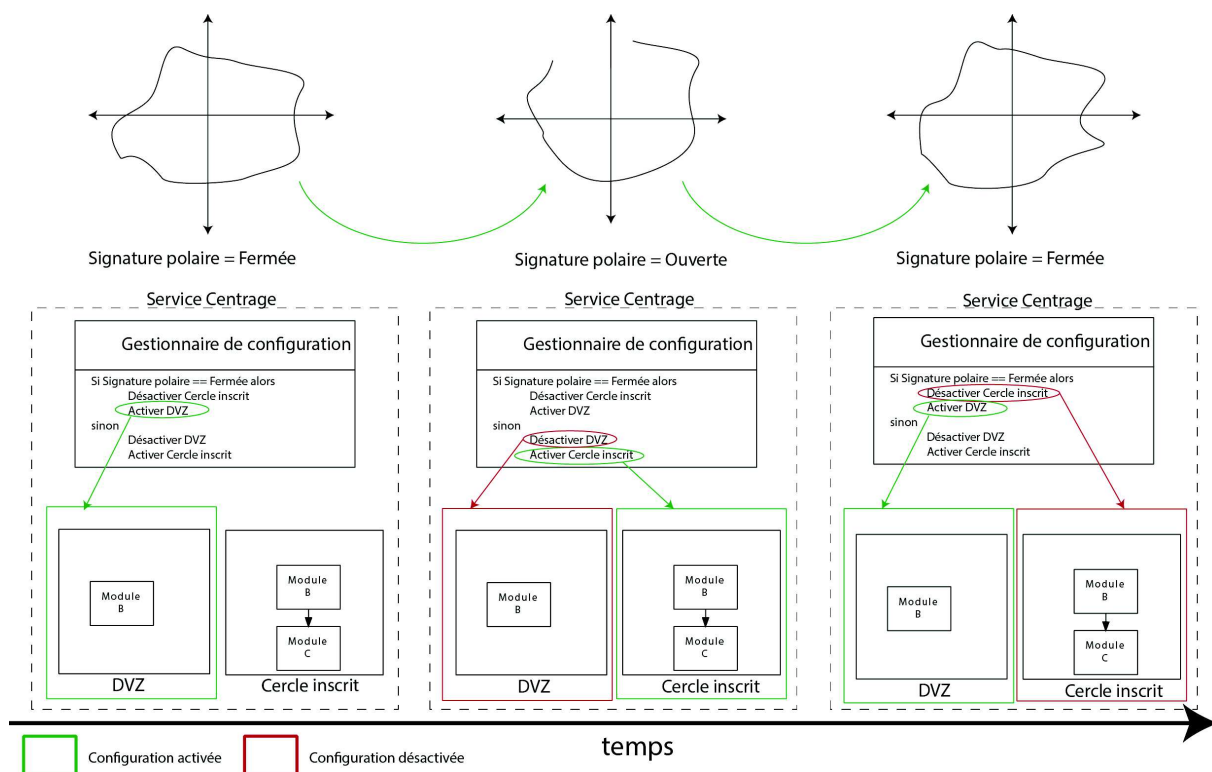


Figure 4-4 : Exemple de changement de configuration lors d'un changement de CA.

La Figure 4-4 permet d'illustrer schématiquement un changement de configuration, dans le contexte de l'exploration karstique, où le sonar profilométrique fournit une « signature polaire », de la coupe du conduit karstique visité. Nous pouvons constater que lorsque nous avons un changement de valeur de la variable d'état « signature polaire », la condition

d'activation est réévaluée afin de réaliser un changement de configuration. La configuration courante, qui exploitait le « centrage DVZ » [85] est alors désactivée (car valide seulement si la signature polaire est fermée) puis remplacée par la configuration adéquate, à savoir celle exploitant le « centrage cercle inscrit » (i.e. pour une signature polaire non fermée)[40].

4.5 Définition de la relation entre ressource, configuration et service

Une *Ressource* est une abstraction d'un élément pouvant être exploité par l'architecture de contrôle du robot. Il existe trois niveaux de ressources :

- Les ressources à *accès libre* : leur nombre d'exploitants n'est pas limité (par exemple, une mesure de capteur peut être transmise à autant d'exploitants que nécessaire).
- Les ressources à *accès contraint* : leur nombre d'exploitants est limité à au plus un (par exemple, un actionnement ne peut recevoir de commandes provenant de multiples sources sinon il y a un risque évident de commandes contradictoires).
- Les ressources *critiques* : ces ressources doivent toujours avoir un et seulement un exploitant (par exemple, un DDL du robot ne peut recevoir que des commandes provenant d'une seule et même source et doit toujours être commandé).

La généralisation du concept de *Ressource*, permet de considérer également comme ressources des entités abstraites (par exemple des données) et pas seulement matérielles (moteur, capteur, etc.). Le service est donc un producteur ou/et un consommateur de ressources comme illustré à la Figure 4-5. L'ensemble des ressources que le service manipule est construit à partir des ensembles de ressources que ses configurations déclarent manipuler.

Un service doit toujours produire les ressources dont il est désigné producteur, quelle que soit sa configuration. Cela est une contrainte essentielle pour la composition de services sur laquelle se fonde l'architecture.

En effet, si un service venait à ne plus produire une ressource utilisée, et donc requise, par un autre service (avec lequel il est composé ou non), alors cet autre service ne pourrait plus être assuré. Cette contrainte est importante sinon un « défaut de production » pourrait remettre en cause toute la composition sur laquelle se base l'architecture (à noter qu'un défaut de production peut être surveillé dans le cadre de la sûreté de fonctionnement, mais ce n'est pas le propos ici).

Un service peut dynamiquement relâcher des ressources qu'il consomme.

En effet, selon la configuration courante un service ne consomme pas nécessairement, à l'instant t , toutes les ressources dont il s'est déclaré potentiellement consommateur. Dès lors les ressources qu'il ne consomme pas peuvent être temporairement relâchées, sans incidence sur les autres services en cours.

C'est pour ces deux raisons que les ressources sont associées aux configurations.

Ceci étant précisé, l'architecture de contrôle est décrite à partir des services et ce sont les services qui sont déclarés, composés, exécutés, etc. De fait, la production et la consommation de ressources sont « affichées » au niveau des services, même si elles sont gérées dynamiquement au niveau des configurations.

De plus, il appartient aux ressources de gérer leurs propres contraintes, émanant de leur niveau d'accès (libre, contraint, critique). Par conséquent la ressource doit connaître à tout instant ses « utilisateurs » en termes de consommateurs, ce qui nécessite donc d'exprimer le lien entre ressource et service.

Il est donc nécessaire de définir un lien entre les services et les ressources, mais ce lien est décrit à travers les configurations puisque ce sont ces entités qui exhibent l'aspect dynamique de l'utilisation des ressources.

De fait, la définition d'une ressource, manipulée par une configuration d'un service (ou de plusieurs selon le niveau d'accès de la ressource), est décrite à travers l'expression (4.5) :

$$R_i \ni \{R_{id}, RT, RL, EL_k\}, \exists! R_{id} \in \{R\}$$

$$EL_k \ni \{L_1, \dots, L_i, \dots, L_j\}, L_i = \{S_{id}\}, \exists! L_i \in \{L\} \quad (4.5)$$

Avec :

- R_{id} , l'identifiant de la ressource R_i , R_i unique et appartenant à l'ensemble des ressources $\{R\}$.
- RT , indique le type de la ressource, i.e. si la ressource est produite ou consommée.
- RL , exprime le niveau (level) d'accès de la ressource (libre, contraint, critique)
- EL_k , exprime l'ensemble des liens entre la ressource R_i et les services. Chaque lien L_i de l'ensemble EL_k désigne une relation entre la ressource R_y et un service précis, identifié par S_{id} . L_i est un lien, unique, appartenant à l'ensemble des liens de l'architecture $\{L\}$.

Connaissant son type, son niveau d'accès et ses liens, il est possible de vérifier la satisfaction des contraintes d'une ressource.

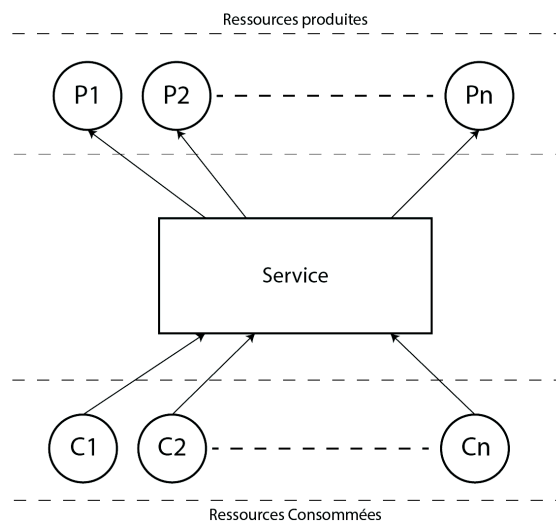


Figure 4-5 : Production/Consommation de ressources par les services

Service, configuration et ressource étant définis, explicitons maintenant la relation entre ces concepts et la « fonction ».

4.6 Définition de la relation entre service, configuration, ressource et « fonction »

Jusque là nous n'avons pas évoqué comment sont associés le service et la « fonction » au sens général du terme (i.e., une tâche robotique, impliquant potentiellement plusieurs traitements).

Nous entendons par « fonction » une relation entrées/sorties, pouvant être d'ailleurs constituée d'un enchaînement de traitements (eux-mêmes assimilables à des fonctions). Cette fonction est exécutée en temps-réel. Sans détailler le middleware temps-réel ContrAct sur lequel est effectuée la projection de l'architecture (cf. section 0), précisons quelques caractéristiques du « modèle d'exécution » :

- Les entités logicielles de base sont les modules.
- Un module encapsule une relation entrées/sorties (fonction).
- Un module possède une interface composée de ports (données, événements, contrôle, paramètres), par le biais de laquelle les modules peuvent être composés et contrôlés.
- Une composition de modules est décrite par un schéma.
- Un schéma « possède » des ports d'entrées/sorties, résultant de la composition des modules qu'il encapsule.
- Un ordonnanceur gère l'ordonnancement de l'exécution des modules des schémas actifs (en respectant l'ordre indiqué par chaque schéma).

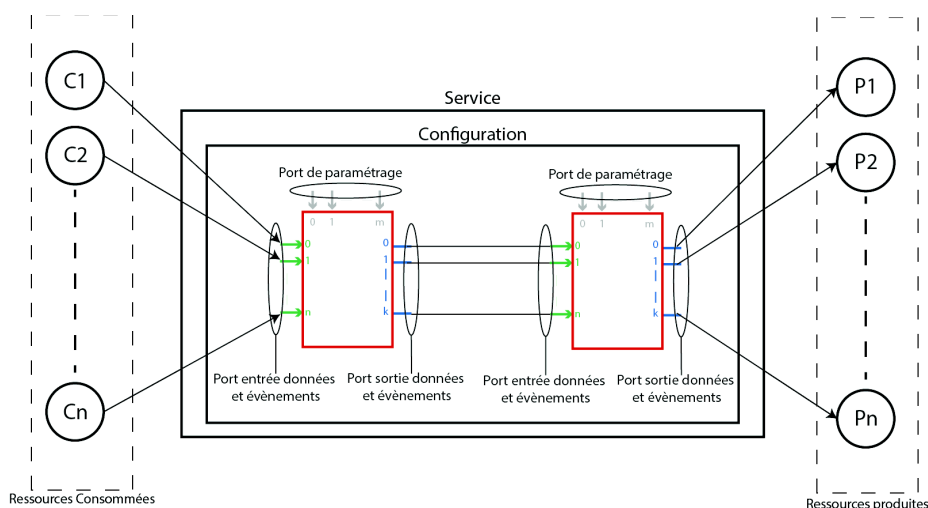


Figure 4-6 : Relation entre service, configuration, ressource et fonction

La Figure 4-6 représente une fonction basée sur la composition d'un ensemble de modules (appelée schéma sous ContrAct), la relation entre la configuration et la fonction, la relation entre service et configuration et enfin la mise en évidence des ressources manipulées (consommées et produites). La notion de ressource a en effet été généralisée et les ports d'entrée et de sortie de la composition (i.e. du schéma) sont considérés, par abstraction, comme des ressources (la « vue » ressource).

Cette abstraction permet de généraliser le concept de ressource et donc la définition du service et la composition de services à travers l'architecture.

Ainsi le lien avec les fonctions (réalisées dans le cœur du service et gérées par le biais de la configuration active) se fait par la production, et la consommation, de ressources (ports des fonctions), ainsi que par le contrôle de leur activité que nous détaillerons à travers l'architecture temps-réel résultante.

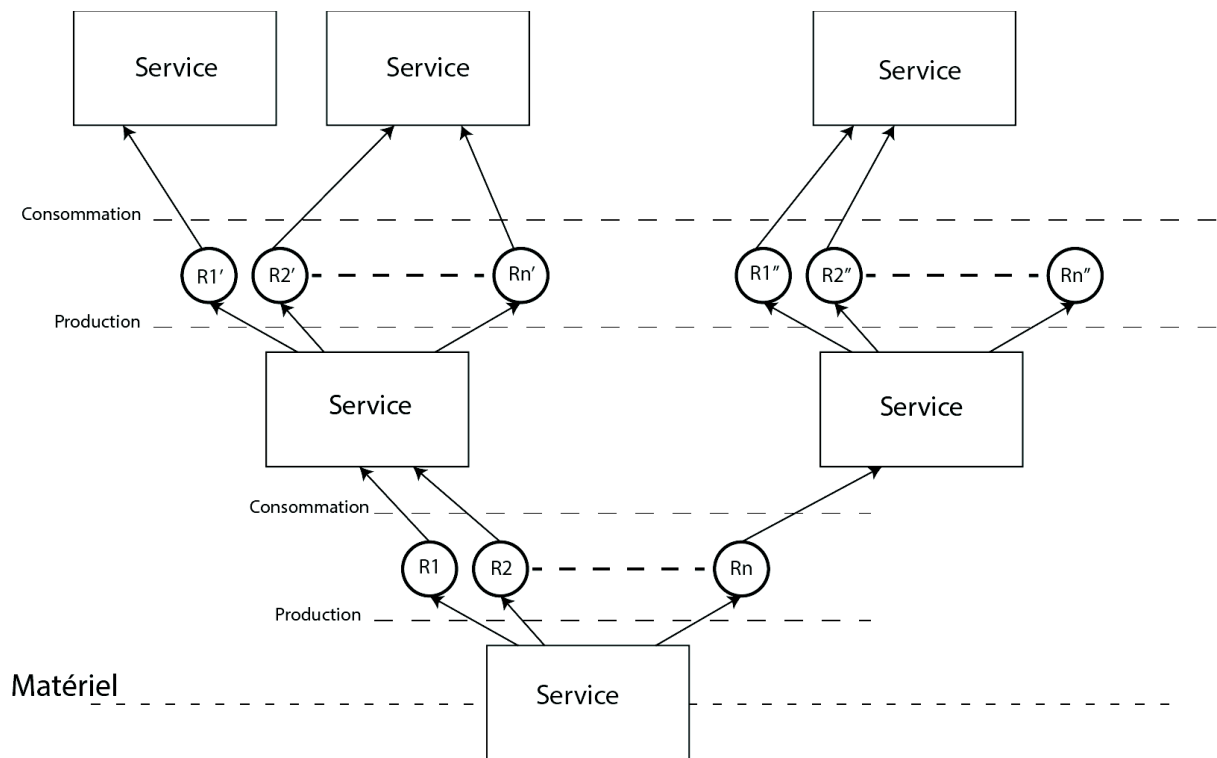


Figure 4-7 : Arbre de services reflétant la composition pour une architecture donnée

Au bilan, cette définition du *Service* et de la *Ressource* permet de construire l'architecture sous la forme d'un arbre où le point de départ est le matériel (Figure 4-7). Le choix du matériel comme point de départ est motivé par l'hypothèse que le service permettant de gérer le matériel (actionnement et capteurs embarqués) ne changera pas en cours de mission (i.e., une fois le vecteur dans l'eau).

Le modèle proposé comprend un ensemble de gestionnaires, pour les services et les configurations. Nous détaillerons plus tard ces gestionnaires en les situant dans l'architecture, en mixant les vues abstraction et exécution (section 4.9).

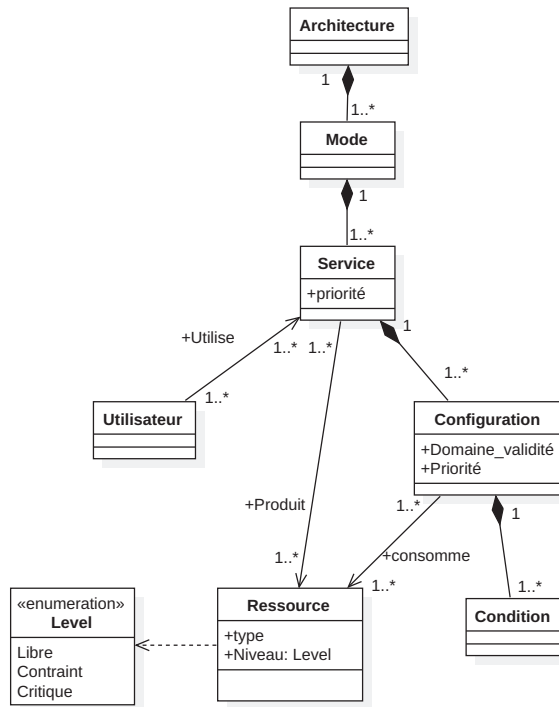


Figure 4-8 : Concepts principaux de l'architecture

L'ensemble des concepts principaux ayant été défini et résumés sur le diagramme UML donné Figure 4-8, nous allons les illustrer à travers des exemples issus du vecteur sous-marin Jack.

4.7 Illustration du modèle d'architecture proposé, pour le robot Jack

Pour chaque concept et entité introduits dans le modèle d'architecture proposé nous allons donner un exemple (simplifié) en considérant le vecteur sous-marin Jack : service, configuration, ressources DDL et ports des modules, etc.

4.7.1 Services et ressources DDL sur le robot Jack

Rappelons les degrés de liberté du vecteur. Dans sa version de base, seulement 5 DDL sont contrôlables sur les 6 possibles, seul le tangage n'est pas motorisé mais il est naturellement stable du fait de son équilibre hydrostatique. L'intégralité des 6 DDL est actionnable dans la version étendue du robot, à savoir la version à 12 moteurs.

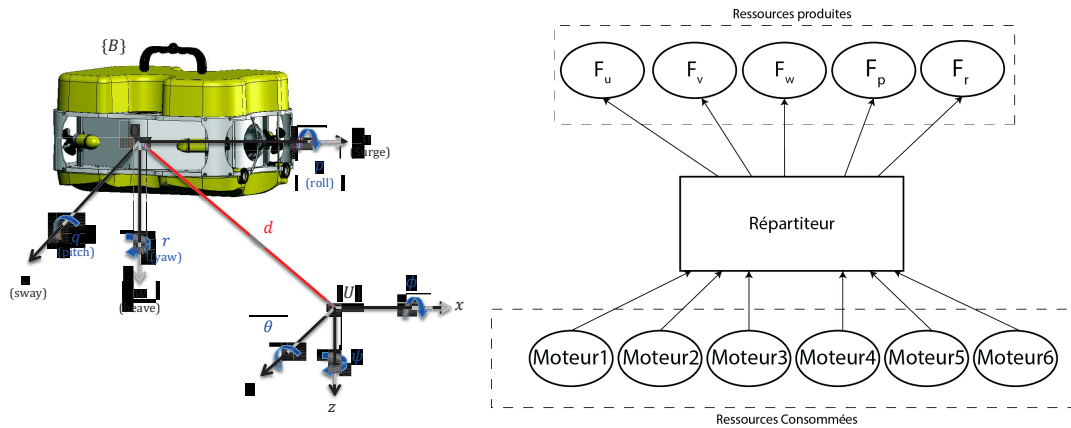


Figure 4-9 : Degrés de libertés « classiques » (DDL) et ressources associées

Spécifier le contrôle selon les DDL vise à rendre le plus transparent possible l'action sur les différents moteurs qui composent l'actionnement du robot. Les différents contrôles (asservissements) portent directement sur les DDL, facilitant la gestion des moteurs au sens de ressources à accès mutuellement exclusifs. Ces DDL ne sont pas directement considérés comme les ressources manipulées ; en effet, nous avons défini une abstraction de l'actionnement (cf. Chapitre 3) à travers laquelle ces DDL sont manipulés en tant que forces requises (par exemple l'action sur le DDL u se décrit par la force F_u).

Prenons l'exemple de l'asservissement en cap. L'asservissement en cap est un service qui influe sur l'ensemble des propulseurs horizontaux pour tenir la consigne opérateur comme cap de référence (Figure 4-10). Le cap est la rotation autour de l'axe z_B (en situation d'attitude nulle pour simplifier les explications) joue sur le DDL nommée F_r . Dans la Figure 4-10, les ressources d'entrée et de sortie de la séquence de modules sont représentées en rouge. Une fois le service activé, le (ou les) DDL concerné(s) n'est (ne sont) plus disponible(s) à d'autres services. Un service résulte, en termes d'exécution, par le séquençage de l'ensemble de modules (schéma) constituant la fonction, comme décrit sur la figure ci-dessous.

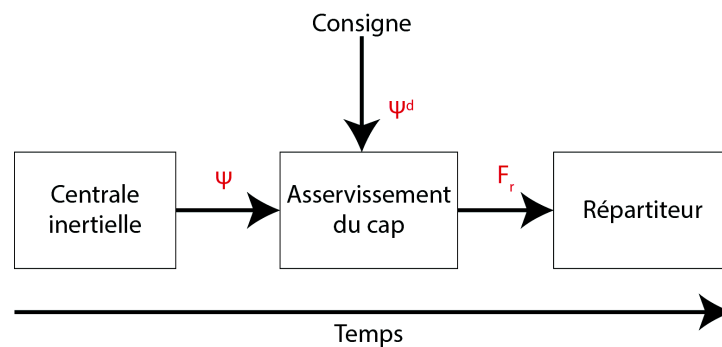


Figure 4-10 : Exemple de fonction (séquence de modules) pour le service d'asservissement en cap

Un autre exemple de service est l'asservissement en profondeur qui est un asservissement en position, i.e. le positionnement automatique du robot à une profondeur donnée sur l'axe

z_B (Figure 4-11). Pour le Jack 6, et considérant une attitude nulle, la ressource F_w va utiliser les deux propulseurs verticaux. Les propulseurs verticaux peuvent également être utilisés par un service de correction automatique du roulis. Le roulis est défini par F_p qui est le couple à appliquer autour de l'axe x_B . La correction du roulis est réalisée grâce à l'action différentielle entre les moteurs verticaux. Dans le cas de l'emport du capteur de vitesse (DVL) (en version 12 moteurs), il est possible de réaliser deux autres asservissements supplémentaires afin d'asservir le système en vitesse sur les axes x_B et y_B .

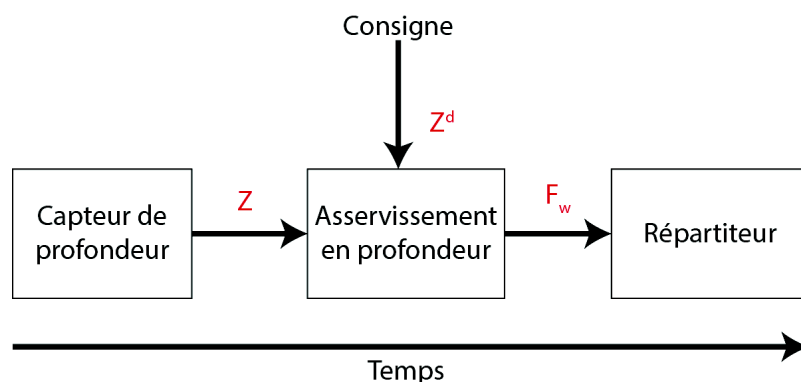


Figure 4-11 : Exemple de fonction (séquence de modules) pour le service d'asservissement en profondeur

Plusieurs services peuvent donc être définis (la liste précédente n'étant pas exhaustive), chacun spécifiant la (ou les) ressource(s) qu'il exploite en termes de DDL qu'il contraint. Cette abstraction des moteurs exploités comme des ressources (les ressources étant les forces qu'ils génèrent) facilite la gestion cohérente des moyens d'actionnement.

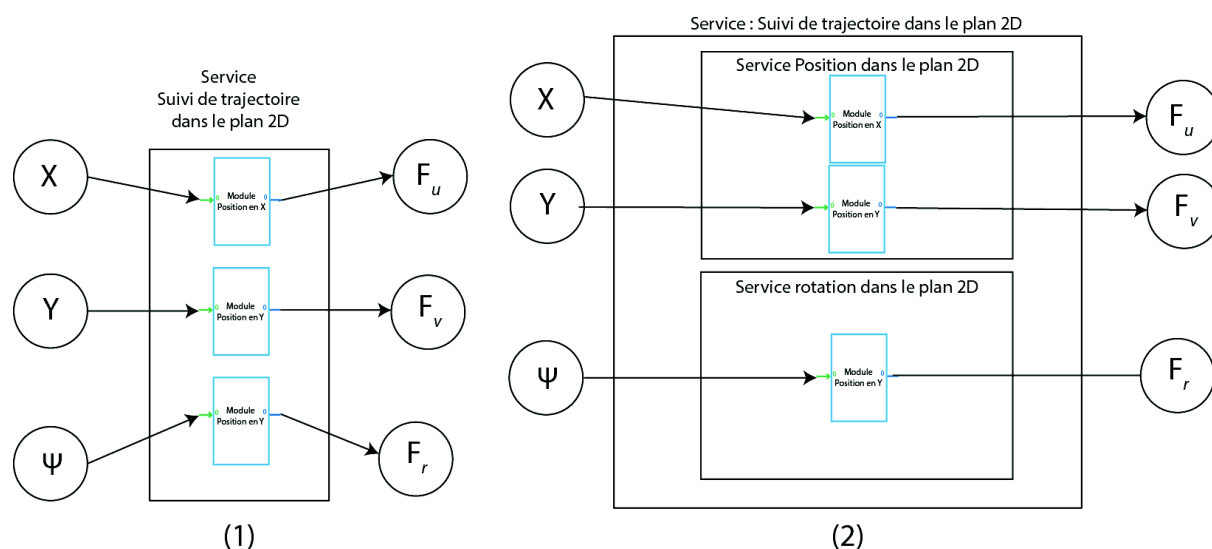


Figure 4-12 : Exemple d'un service évolué

Précisons que les services (de base) peuvent éventuellement être composés pour définir des services plus « évolués », et donc qu'un même service peut éventuellement exploiter plusieurs DDL. Il existe parfois plusieurs façons de décrire un service « évolué ». Par exemple, un suivi

de trajectoire dans le plan 2D peut se définir comme un seul et même service de suivi de trajectoire exploitant les DDL ψ , X et Y (Figure 4-12(1)), et agissant donc sur les ressources F_r , F_u et F_v , ou un service composé de deux services de base (Figure 4-12(2)), l'un gérant uniquement les DDL X et Y (agissant donc sur les forces F_u et F_v) et l'autre gérant le cap de l'engin via le DDL ψ (agissant sur la force F_r).

4.7.2 Services et ressources capteurs sur le robot Jack

L'étude serait incomplète si on ne considérait pas les capteurs, en particulier pour le contexte sous-marin. En effet, en robotique sous-marine les capteurs utilisés sont souvent des capteurs acoustiques; or, selon leurs caractéristiques techniques, ils peuvent potentiellement interférer entre eux (gamme ou harmonique de fréquence identique). De fait, leur recrutement, au sens de leur utilisation, peut devoir répondre à cette contrainte d'exclusion mutuelle ou de séquençement dans leur recrutement. Cette problématique a été traitée dans l'équipe à travers la thèse de A. El Jalaoui [86].

Distinguer les capteurs c'est aussi distinguer le type de capteurs au sens de capteurs motorisés ou non ; à titre d'exemple la caméra embarquée est pilotable. Dès lors ces capteurs doivent également être pris en compte comme des ressources sur lesquelles un seul mode de fonctionnement peut être imposé à la fois. Dans le cas de la caméra embarquée par exemple, elle peut être soit directement télé-opérée par l'opérateur, soit être impliquée dans une commande référencée vision ; un seul de ces modes de fonctionnement ne peut être actif à un instant donné. Au delà, certains capteurs dépendent aussi parfois de DDL : une caméra équipée d'un pan tilt dispose par exemple de deux degrés de libertés (θ et ψ). Cela sera géré de la même manière que les ressources DDL d'actionnement précédemment évoquées. La Figure 4-13 est un service constitué d'une configuration permettant de produire les ressources relatives à la gestion de la caméra (flux vidéo et position). Les entrées du module permettant de gérer le pan et le tilt (θ et ψ) correspondent aux ressources θ_{cam} et ψ_{cam} et la ressource Cam correspond aux données de sortie du module caméra.

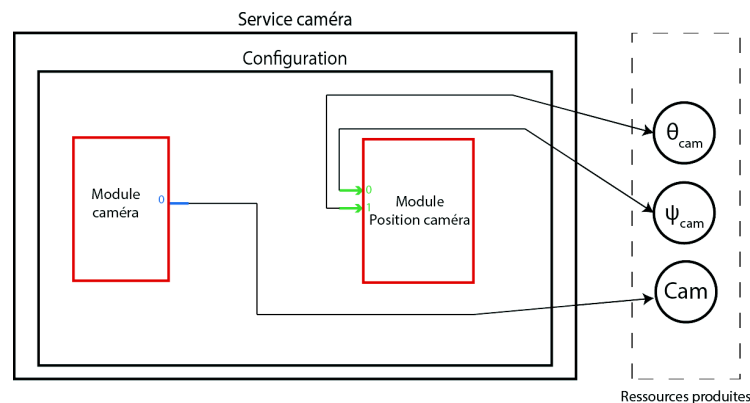


Figure 4-13 : Représentation schématique du service « Caméra » et des ressources manipulées

4.7.3 Configurations et fonctions

Un service peut avoir plusieurs configurations possibles, une seule active à la fois. A chaque configuration est associée la fonction correspondante. L'ensemble des conditions d'activation permet d'exprimer la sélection dynamique de la configuration adéquate selon le contexte (comme évoqué section 4.4). La figure ci-dessous montre un service contenant 2 configurations différentes, chacune ayant une fonction associée (la fonction étant l'ensemble de modules « encapsulé » par la configuration sur la figure). Les ressources manipulées par le service est l'ensemble des ressources manipulées par ses configurations. Ce service a pour fonction de faire un asservissement du cap (ψ) par rapport à la consigne (ψ^D), la première configuration utilise une centrale inertielle pour mesurer le cap (ψ) et la vitesse de rotation autour de l'axe $z_B(r)$, et la seconde configuration réalise une dérivation de l'erreur afin d'obtenir une vitesse estimée de la rotation autour de z_B . Cette deuxième configuration est une alternative dans le cas où le robot n'est pas équipé d'une centrale inertielle mais qu'il dispose uniquement d'un magnétomètre pour la mesure du cap (ψ). La condition d'activation (« Centrale inertielle ») se fait donc sur la présence ou non de la centrale inertielle (absente de la version de base du Jack).

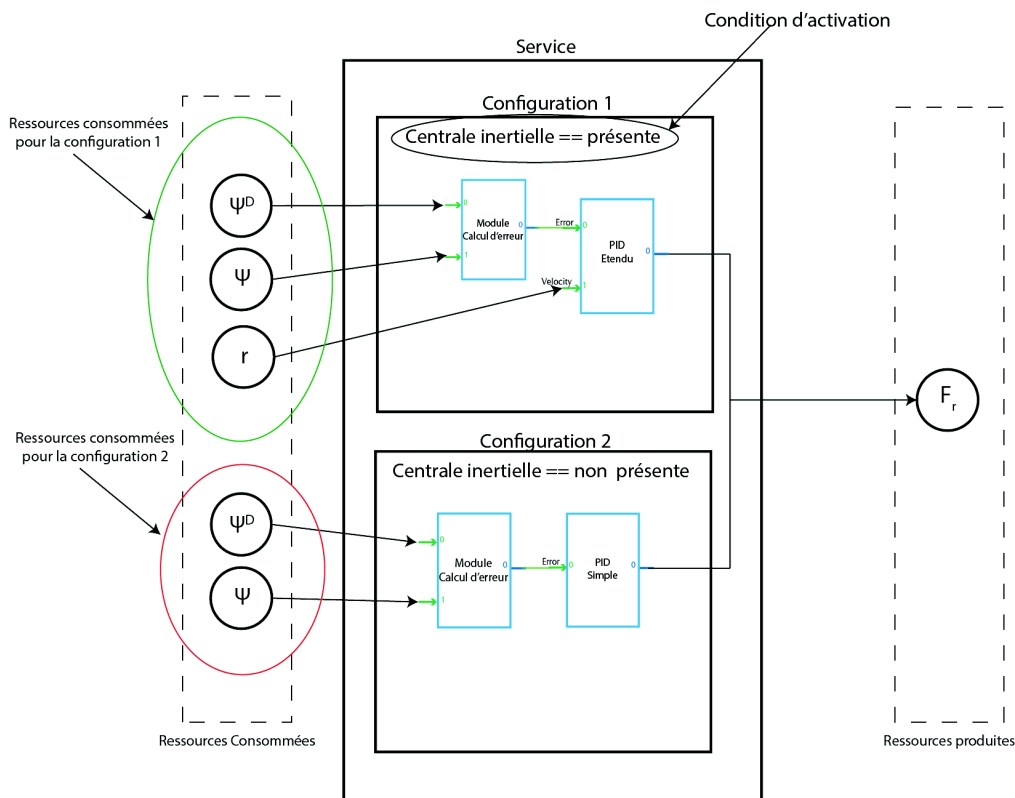


Figure 4-14 : Exemple de service ayant 2 configurations

4.7.4 Changement automatique de configuration

Reprenons l'exemple vu précédemment (décrit Figure 4-15), pour illustrer une commutation de configuration de façon automatique. Lorsque le système ne dispose pas de centrale inertielle, la configuration change de façon automatique pour adapter la loi de contrôle afin de considérer une estimation de la vitesse de rotation (r), plutôt que sa mesure. Nous pouvons observer que les ressources consommées sont différentes d'une configuration à

l'autre, lorsque le changement de configuration est réalisé les ressources requises par la configuration antérieure sont libérées et les nouvelles ressources nécessaires pour la nouvelle configuration sont acquises. Notons que les ressources produites sont identiques (en l'occurrence sur cet exemple la ressource Fr), même si leur production ne relève pas de la même configuration après la commutation.

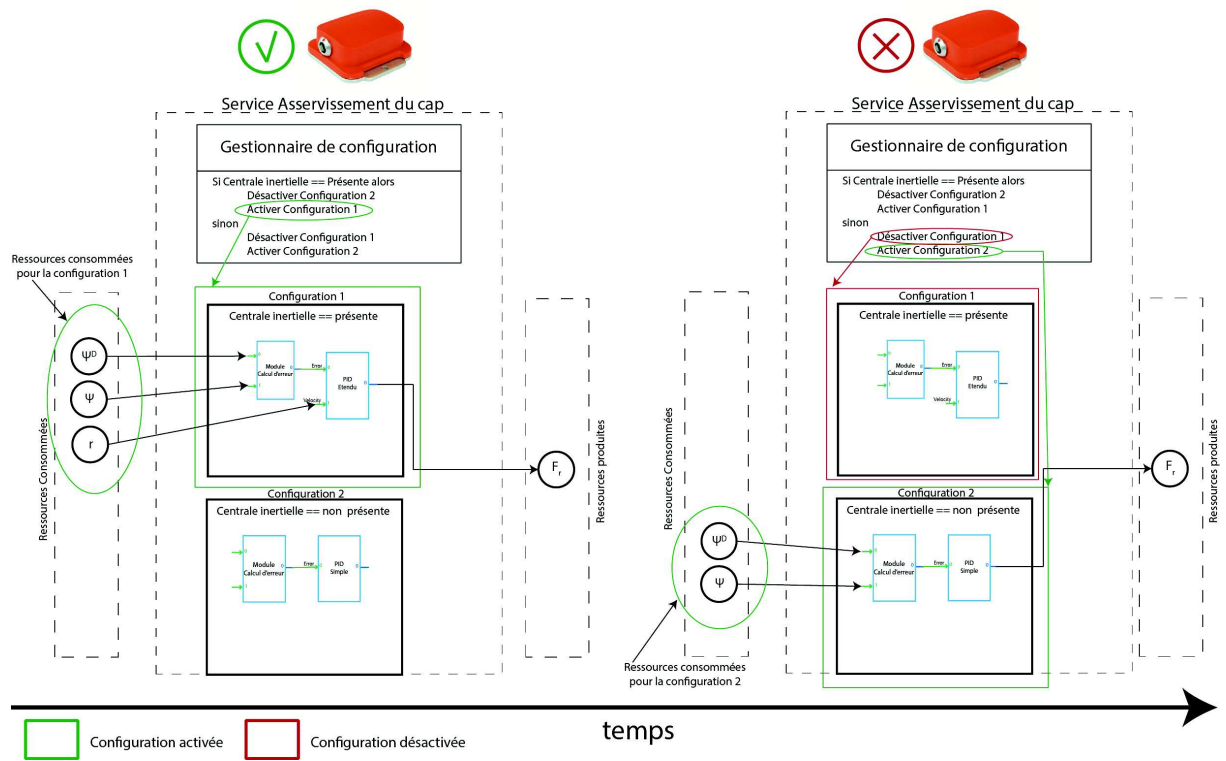


Figure 4-15 : Exemple de commutation automatique de configuration

Nous allons maintenant détailler comment le modèle architectural et les concepts que nous avons décrits ont été projetés sur le middleware temps-réel *ContrAct*, c'est à dire comment notre proposition a été concrétisée pour créer l'architecture temps-réel embarquée du vecteur sous-marin *Jack*.

Pour ce faire, nous allons d'abord présenter le middleware *ContrAct*.

4.8 Le middleware ContrAct

L'architecture logicielle est projetée sur le middleware temps-réel, développé par le LIRMM, ContrACT [38], [39], implémenté en l'occurrence sur le système temps-réel Xenomai (une extension libre du noyau Linux lui apportant des fonctionnalités temps-réel).

ContrACT (Control Architecture Creation Technology) permet la conception et le développement d'architectures de contrôle. La structuration architecturale n'est pas imposée mais elle est basée sur le concept de couche (niveau). La Figure 4-16 illustre ces 2 couches :

- la couche exécutive responsable de la réalisation des décisions prises dans la couche supérieure, à savoir de l'exécution des fonctions.
- la couche décisionnelle qui contient les processus liés à la prise de décisions, elle définit les réactions à adopter en fonction des événements provenant de la couche inférieure.

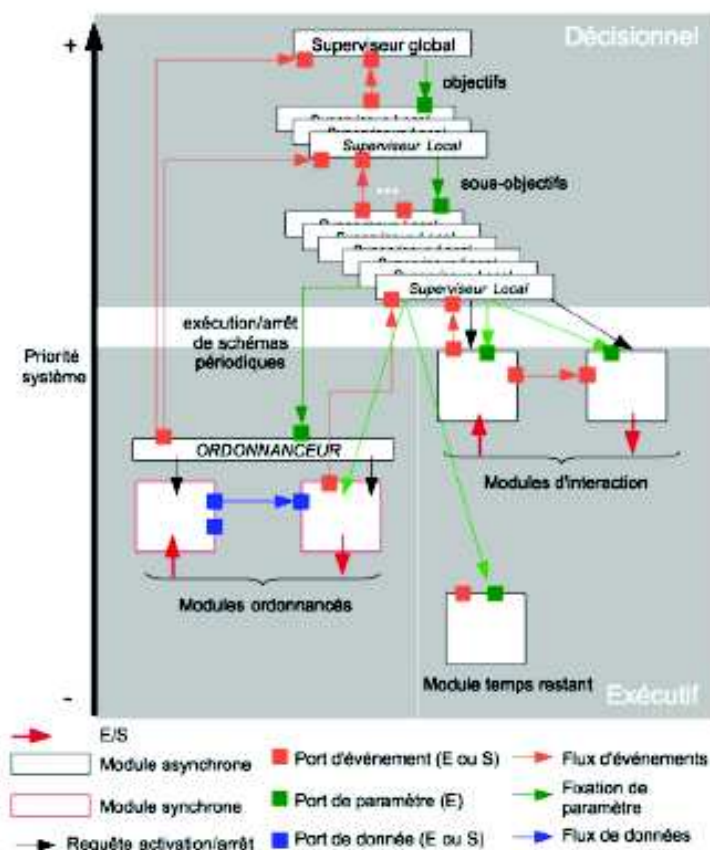


Figure 4-16 : Modèle architectural [39]

Afin d'organiser les processus au sein de ces deux couches, le modèle de programmation repose sur le concept de module (Figure 4-17), défini comme une tâche temps-réel indépendante qui réagit à un ensemble de requêtes prédéfinies et qui communique avec les autres par le moyen de ports. Les ports sont typés : des ports de données, des ports

d'événements et des ports de contrôle et de paramétrage. Les modules sont composés par l'intermédiaire des ports ; la composition peut être statique (définie avant l'exécution) ou dynamique (lors de l'exécution).

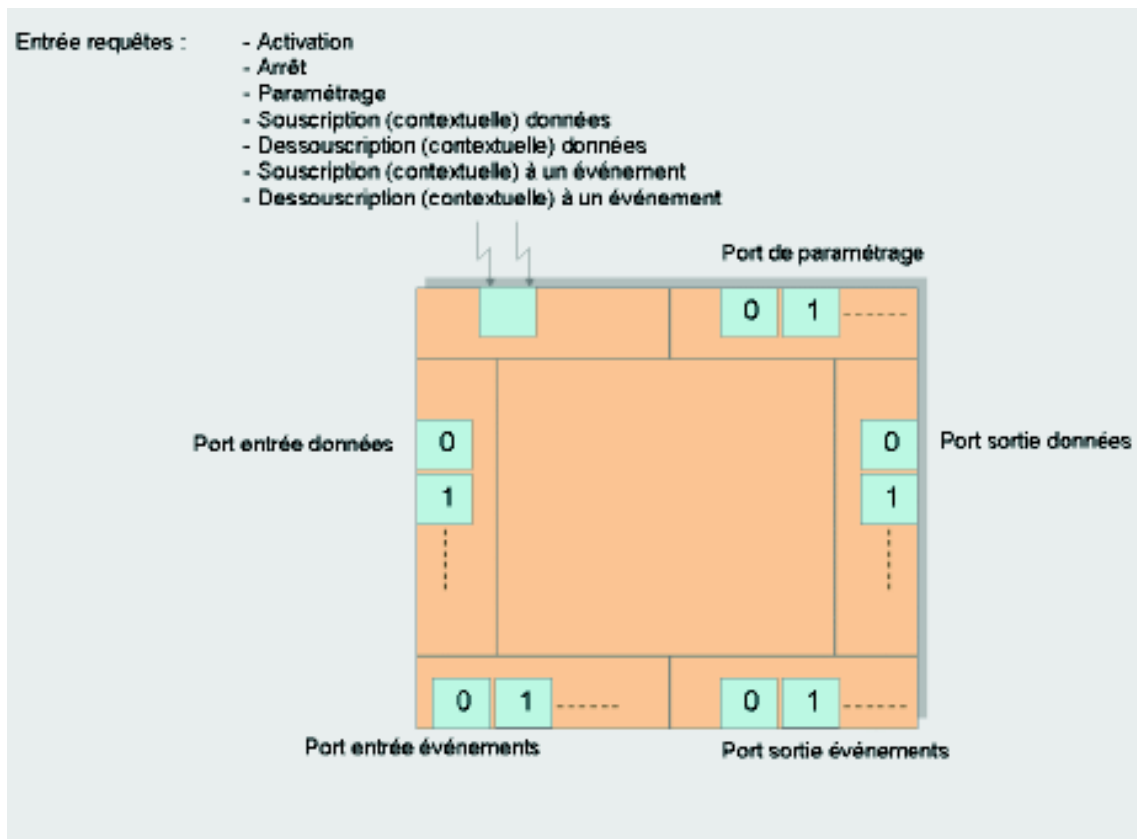


Figure 4-17 : Modèle de module ContrAct [38]

Les modules peuvent être classifiés au sein des deux niveaux d'interaction définis précédemment. Le niveau décisionnel contient les modules de supervision, de nature asynchrone. Le niveau exécutif contient les modules de nature synchrone ainsi que le module spécifique d'ordonnancement. L'utilisateur va pouvoir décrire, une fois les modules déclarés, ce que l'on appelle des schémas. Ces schémas représentent une composition ordonnée de modules s'exécutant à une période donnée. Au sein des modules de supervision, l'utilisateur définit des règles auxquelles sont associées une condition d'activation et de désactivation (conditions sur le temps d'exécution, sur la réception d'événements entre autres). Il a la possibilité au sein de ces règles d'activer des schémas, de faire dynamiquement des liaisons de données entre schémas, etc ...

Comme nous l'avons mentionné, tous les modules sont des tâches pour le système d'exploitation temps-réel mais différentes classes sont dérivées du modèle précédent (Figure 4-17):

- L'ordonnanceur, un module unique implémentant les algorithmes d'ordonnancement.
- Les modules de supervision, permettant d'implémenter des règles de supervision, et autres modules asynchrones implémentant les réactions spontanées à des événements.

- Les modules synchrones, implémentant les algorithmes de calculs périodiques ainsi que l'échantillonnage des capteurs et la commande des actionneurs.
- Les modules « temps-restant » (i.e., exécutés lorsque le processeur n'est pas utilisé par les modules synchrones, donc jusqu'à la prochaine échéance de l'ordonnanceur), implémentant les calculs lourds lorsque le processeur est libre d'utilisation.

La génération de cette architecture se fait via l'utilisation d'un environnement de développement qui automatise le processus de développement et qui le rend aussi fiable que possible. La programmation est basée sur un langage nommé Cactal qui permet de décrire facilement les modules, les comportements de superviseurs et les connexions entre les différents ports [87].

La conception de modules est couplée au concept d'atome, une notion développée dans le cadre de la thèse de Adrien Lasbouygues [40]. Un atome (Figure 4-18) est un objet visant à représenter une entité avec ses entrées, ses sorties et sa physique interne. Chaque module a au moins un atome qui lui est associé. Ces atomes sont centralisés au sein d'une librairie système. Ce concept nous permet de limiter les erreurs au niveau du typage de données, des liaisons inter-atomes, et de la physique interne, et a pour conséquence directe de faciliter les développements, ainsi que de contribuer à la modularité, la réutilisabilité, l'évolutivité de l'architecture [88]. Nous ne détaillerons pas ces travaux, nous précisons simplement que c'est ainsi qu'ont été développés nos modules.

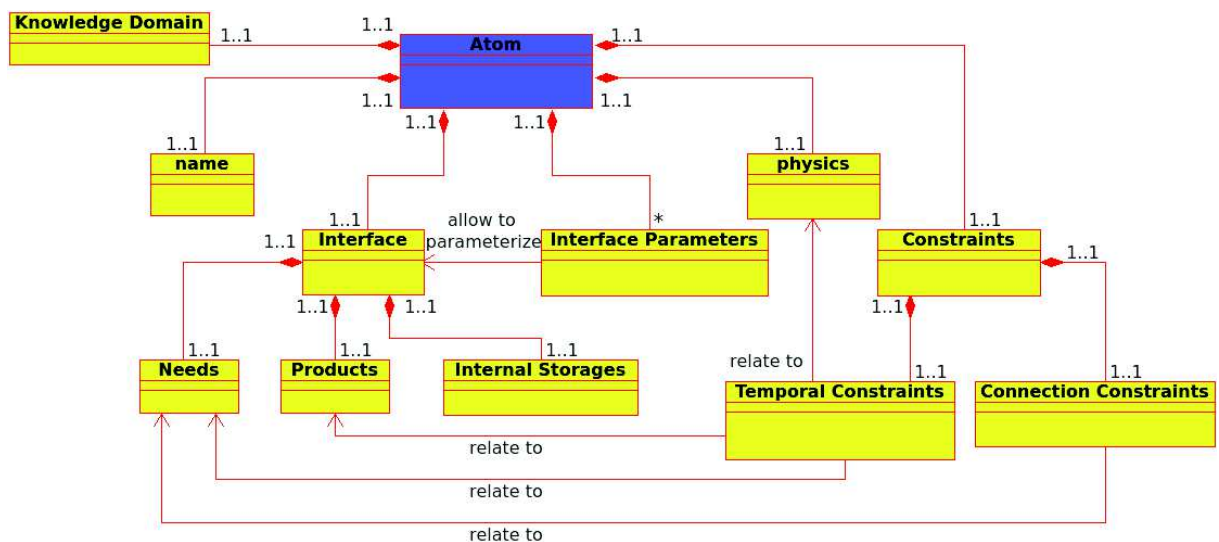


Figure 4-18 : Description d'un atome [40]

Les principes et mécanismes du middleware temps-réel ContrAct ayant été présentés, nous allons décrire la projection de notre architecture sur ContrAct, et expliciter notamment les gestionnaires de services et de configurations.

4.9 Projection de l'architecture sur le middleware ContrACT.

La projection de notre architecture sur le middleware temps-réel ContrAct a été brièvement évoquée à travers les sections précédentes mais donnons ici une vue globale, situant les gestionnaires mis en place pour l'approche à service.

4.9.1 Structuration de l'architecture globale sur ContrAct.

La Figure 4-20 illustre la structuration de notre architecture selon 2 niveaux, en l'occurrence les couches décisionnelle et exécutive.

La gestion de services est déployée dans le niveau décisionnel. Deux modules spécifiques, asynchrones, sont déployés : le gestionnaire de service et le module de communication avec l'opérateur. Le gestionnaire de service communique avec le module de communication (événementielle) par le biais des ports à événements. Lorsque le module de communication reçoit une notification (message) de l'interface opérateur celui-ci la transmet directement au gestionnaire de services. Par exemple, si l'opérateur demande l'activation d'un mode particulier, le module de communication transmet directement l'ordre d'activation du mode au gestionnaire de services qui prend en charge le changement de modes, à savoir le changement de services. L'activation et la désactivation des différents services sont réalisées avant que les gestionnaires de configuration du niveau exécutif ne se chargent de réaliser la sélection de la configuration la plus adaptée au contexte (pour les services à activer). Une fois réalisé le choix des configurations les plus adaptées au contexte, le gestionnaire de services réalise les différentes connexions entre les modules concernés des différents services.

La gestion des configurations est assurée au sein du niveau exécutif par le biais de superviseurs, chaque service ayant son superviseur (appelé superviseur de service). Pour chaque service, les configurations sont décrites à partir de règles que le superviseur exécute (Figure 4-19-gauche). Lorsque le superviseur réalise des changements de configuration, i.e. lorsqu'une règle est exécutée, il le notifie par événement au gestionnaire de services, du niveau décisionnel (Figure 4-19-droite).

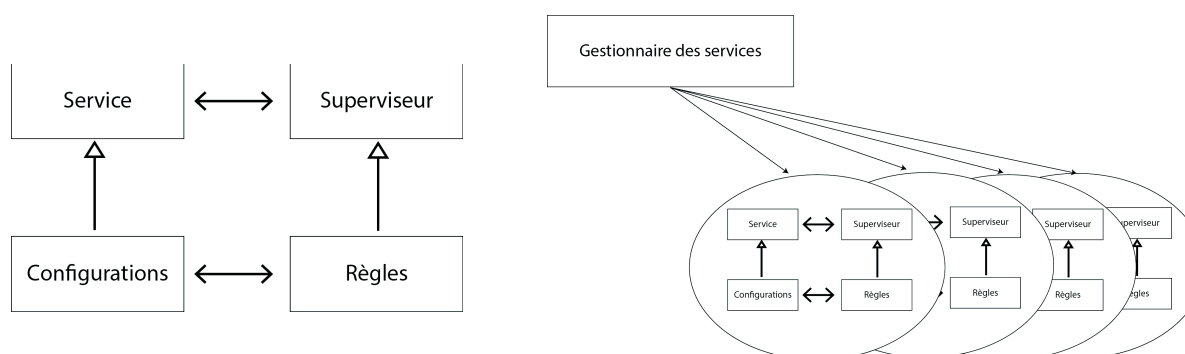


Figure 4-19 : Structure du service (gauche) et du gestionnaire des services (droite) sur ContrACT

Les superviseurs de services configurent les fonctions, au sens du paramétrage des modules composant lesdites fonctions. En effet, nous avons précédemment mentionné que les fonctions sont mises en œuvre par le biais de schémas, eux-mêmes encapsulant une composition de modules (correspondant à la relation entrées/sorties de la fonction concernée).

L'exécution des modules, selon les contraintes de précedence définies dans les schémas, est à la charge de l'ordonnanceur.

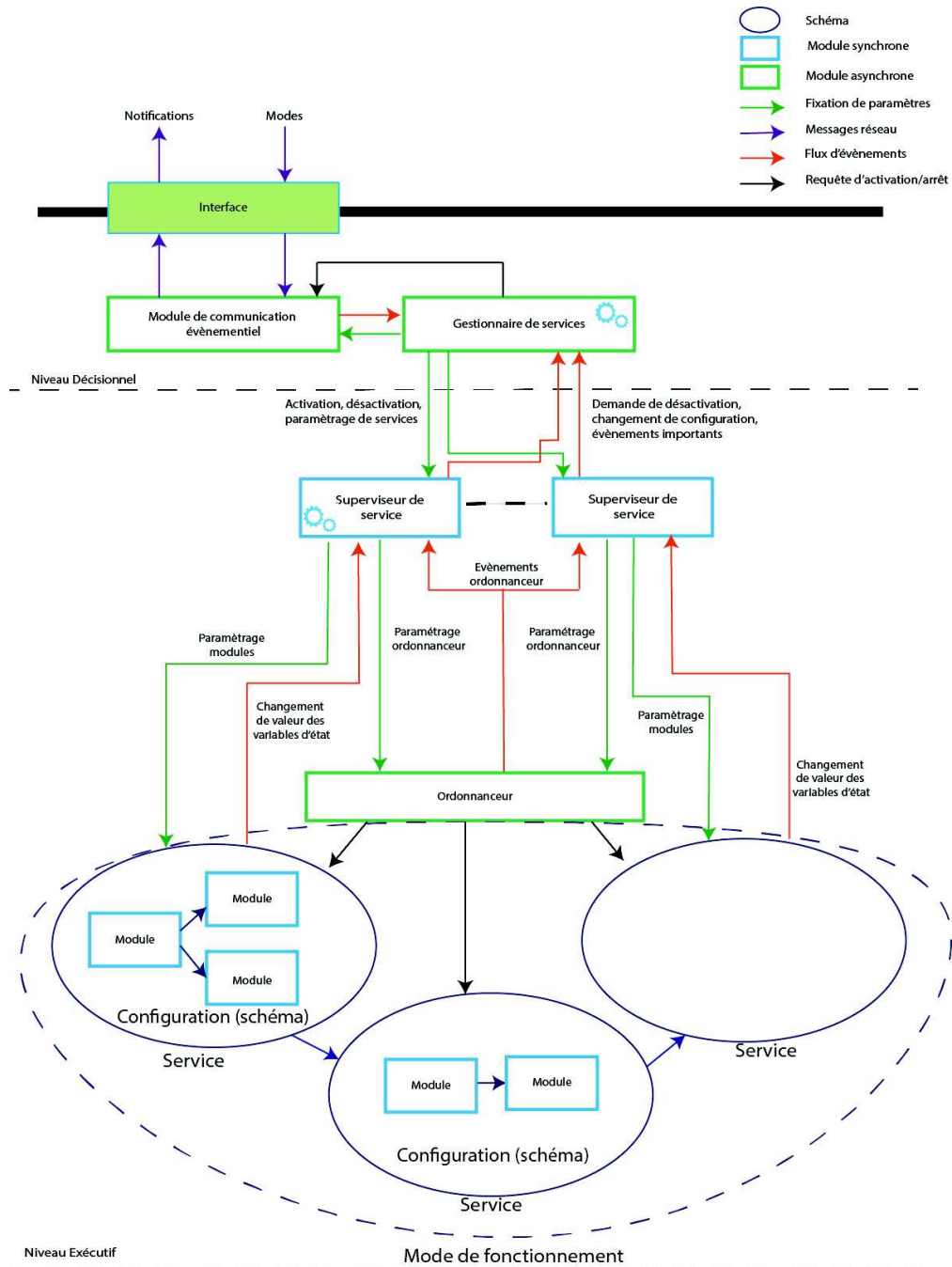


Figure 4-20 : Projection de l'architecture globale sur le middleware ContrAct

Le diagramme de séquence (Figure 4-21) illustre la séquence des différentes opérations réalisées par le système lors d'un changement de mode de fonctionnement.

Rappelons que l'architecture est, pour l'opérateur, structurée en différents modes. Un mode consiste en un ensemble de services. Dans le cas de notre architecture, nous avons défini à ce stade un mode de télé-opération du robot, un mode d'asservissement en cap, un mode d'asservissement en profondeur et un mode regroupant les deux contrôles précédents. Lors d'un changement de mode, le gestionnaire de services va tout d'abord analyser que le changement est possible (en considérant notamment les ressources). Il va analyser les

changements à réaliser et va activer/désactiver les services nécessaires en passant par les superviseurs concernés.

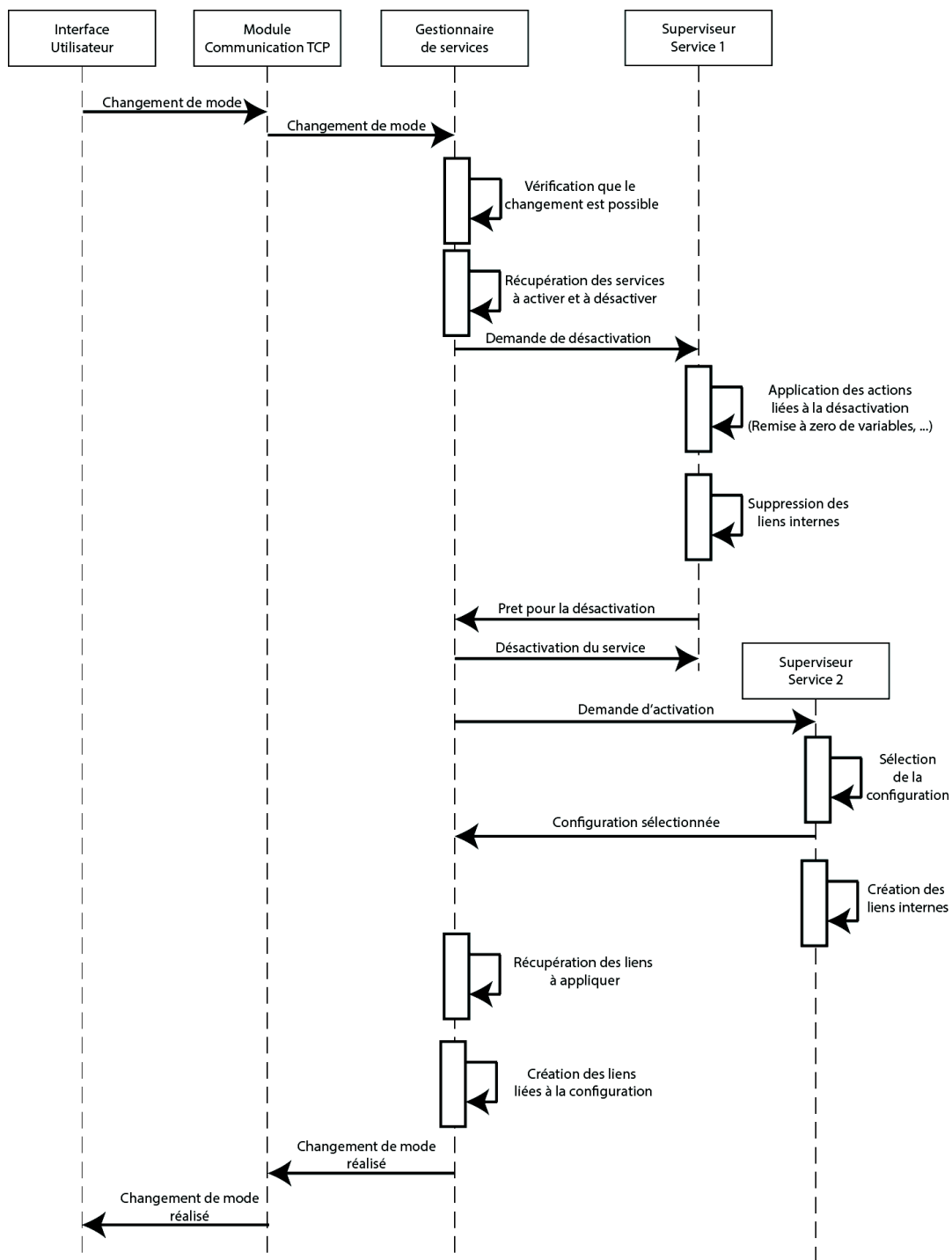


Figure 4-21 : Diagramme de séquence d'un changement de mode de fonctionnement

En complément de la séquence décrite sur la Figure 4-21, la Figure 4-22 donne une représentation schématique du séquençage qui est réalisé lorsque le gestionnaire de services

lance des activations de services. La Figure 4-22(a) est le cas initial lorsque les deux services (Service(1) et Service(2)) ne sont pas actifs. Nous prenons comme exemple le cas d'un opérateur qui fait la demande d'activation d'un service (par exemple le service(1)). Le service(1) se met alors en attente d'un retour de son superviseur qui va lui indiquer quelle est la configuration correspondant au domaine d'activation courant. Dans le schéma (b) de la figure, les deux services sont activés et sont en attente de notification des configurations les plus adaptées, choisies par les superviseurs des deux services concernés. Cette attente est illustrée en bleu dans (b) et (c). Une fois que le superviseur de service a sélectionné la configuration, celui-ci l'indique au gestionnaire de services qui réalisera alors les différents liens nécessaires pour que les services fonctionnent entre eux (comme indiqué en vert dans (c) et (d)).

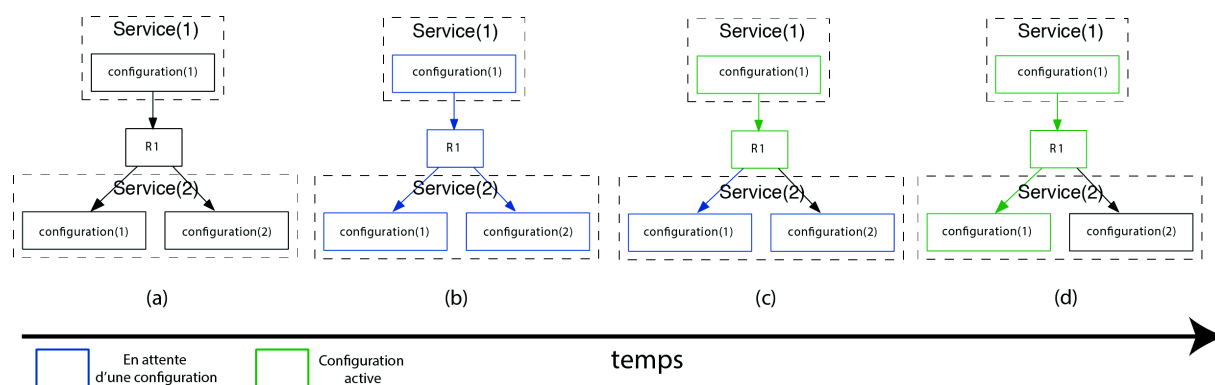


Figure 4-22 : Exemple d'Activation de services

La description de l'architecture est faite par une description, au sein de fichiers, des modes et des services de l'application. Nous allons exposer comment sont structurées ces informations, au regard du middleware retenu.

4.9.2 La gestion de service : descriptions basées XML

Le gestionnaire de service est un mécanisme décisionnaire qui permet de faire un changement de mode de fonctionnement lorsque le robot est en cours de mission. Ce gestionnaire de service permet l'activation, le paramétrage et la désactivation des services nécessaires ou non à l'activation d'un mode. L'architecture globale, les modes de fonctionnement, les services et les ressources sont définis par l'intermédiaire d'un ensemble de fichiers XML. Par exemple, les extraits de fichiers XML donnés Figure 4-23 et Figure 4-24 contiennent respectivement la description de l'architecture globale et les différents modes de fonctionnement.

```

<?xml version = "1.0" encoding="UTF-8" standalone="yes" ?>
<!DOCTYPE Architecture SYSTEM "Architecture.dtd">
<Architecture xmlns:xi="http://www.w3.org/2001/XInclude">

  <xi:include href="ServiceLog.xml"/>
  <xi:include href="ServiceDrivers.xml"/>
  <xi:include href="ServiceDispatcher.xml"/>
  <xi:include href="ServiceModelDynamic.xml"/>
  <xi:include href="ServiceNavigation.xml"/>

  ...

</Architecture>

```

Figure 4-23 : Extrait du fichier XML décrivant l'architecture globale

```

<?xml version = "1.0" encoding="UTF-8" standalone="yes" ?>
<!DOCTYPE Architecture SYSTEM "Modes.dtd">
<Modes xmlns:xi=""http://www.w3.org/2001/XInclude">

  <!-- mode initial avec les joysticks en forces-->
  <Mode
    Id = "1"
    Name = "Default"
    Priority = "1">

    <xi:include href="ServiceDrivers.xml"/>
    <xi:include href="ServiceDispatcher.xml"/>
    <xi:include href="ServiceNavigation.xml"/>
    <xi:include href="ServiceMapping_X_Force.xml"/>
    <xi:include href="ServiceMapping_Y_Force.xml"/>
    <xi:include href="ServiceMapping_Z_Force.xml"/>
    <xi:include href="ServiceMapping_PHI_Force.xml"/>
    <xi:include href="ServiceMapping_PSI_Force.xml"/>
    <xi:include href="ServiceLog.xml"/>
  </Mode>

  <!--Autre mode-->
  <Mode
    Id = "2"
    Name = "Default"
    Priority = "1">
    ...
  </Mode>

</Modes>

```

Figure 4-24 : Extrait du fichier XML décrivant les modes de fonctionnement

La Figure 4-25 représente l'arborescence de l'architecture globale. Cette arborescence, construite à partir des fichiers XML, est la description de l'ensemble des services qui compose l'architecture.

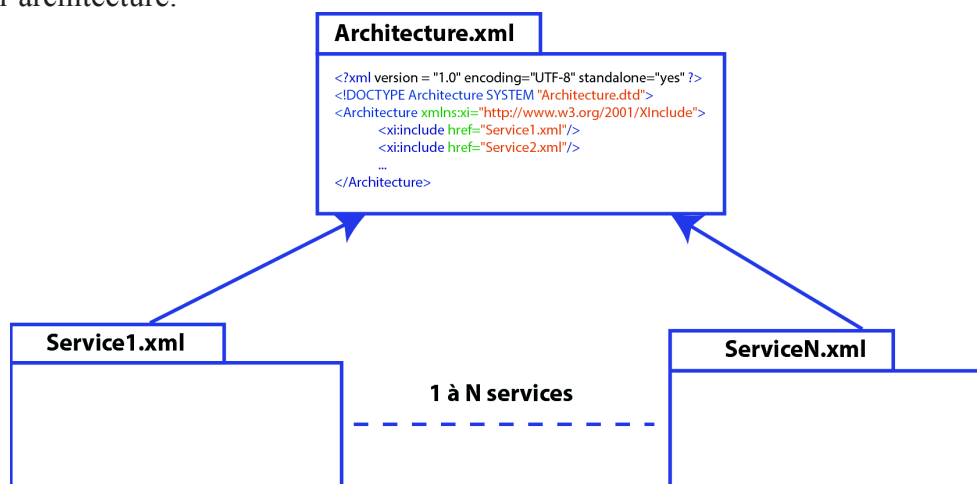


Figure 4-25: Arborescence XML de l'architecture

Chacun des services est décrit dans un fichier XML, ce fichier est constitué de balises permettant de décrire le service (Figure 4-26). La balise <Service> requiert les paramètres suivants :

- L'Id est l'identifiant unique, c'est un paramètre permettant d'identifier le service dans les différentes phases d'élection des services nécessaires dans un mode de fonctionnement.
- Le nom et la description sont des éléments permettant de faciliter le développement de l'architecture en apportant des éléments de précisions à la personne décrivant l'architecture.
- La priorité est utilisée lorsque plusieurs services sont utilisables dans le même contexte, le gestionnaire de service élira le service avec la priorité la plus élevée dans l'architecture (selon une priorité décroissante, i.e. la priorité 1 est la plus élevée).
- Le groupe est associé à l'utilisateur, si l'utilisateur n'est pas autorisé à réaliser des actions sur le service, les changements désirés par l'utilisateur ne seront pas réalisés.
- Le superviseur est le nom du superviseur ContrAct associé au service, c'est ce superviseur qui va réaliser les différents changements de configuration.
- Le paramètre «switchMode» permet à un service de faire une commutation de mode de fonctionnement lorsque le système a réalisé certaines actions, par exemple dans un centrage dans un conduit karstique. Lors d'un centrage, le système nécessite un certain nombre de mesures sonar valides avant de pouvoir s'activer, pour ce faire nous avons besoin d'un mode qui attend le nombre de points nécessaires, une fois que la condition est réalisée le système basculera de lui-même dans le mode fonctionnement exploitant les points sonar.

```

<Service
  Id = "2"
  Name = "Exemple1"
  Description = "Exemple1"
  Priority = "1"
  Groupe = "system"
  Supervisor = "SE1"
  SwitchMode = "True "
>
...
</Service>

```

Figure 4-26 : Extrait d'un fichier XML de définition d'un service

La description du service dispose d'un ensemble de configurations délimité par la balise <Configurations>. Chacune des configurations (<Configuration>) du service dispose d'un certain nombre de paramètres (Figure 4-27):

- L'id est la clé unique permettant de différencier chacune des configurations d'un même service et ainsi éviter qu'aucune configuration ne dispose du même identifiant au sein d'un même service.
- Le nom et la description sont des paramètres permettant aux développeurs travaillant sur l'architecture de mieux documenter ces définitions de service.
- Le paramètre «action» est le nom de la règle du superviseur à appliquer lors de l'activation de la configuration. Cette règle permet au port d'évènement de notifier au gestionnaire de service qu'un changement de configuration a été réalisé et que le gestionnaire de service doit réaliser les différents liens associés à cette nouvelle configuration activée.

```

<Configurations>
  <Configuration
    Id="1"
    Name="exemple0_conf1"
    Description = "exemple_conf1"
    Action="CONF1">
    <Ressources>
      ...
    </Ressources>
  </Configuration>
  <Configuration
    Id="2"
    Name="exemple0_conf2"
    Description = "exemple_conf2"
    Action="CONF2">
    <Ressources>
      ...
    </Ressources>
  </Configuration>
</Configurations>

```

Figure 4-27 : Extrait d'un fichier XML de définition des configurations d'un service

Chacune des configurations est constituée d'une liste de ressources exploitées ou créées par la configuration avec comme contrainte que toutes les configurations d'un même service produisent et requièrent des ressources identiques dans chacune des configurations (sinon cela remettrait en cause une éventuelle composition de services). Une ressource est définie par les paramètres suivants (Figure 4-28):

- L'Id permet au gestionnaire de services d'identifier les ressources. Cet Id est lié à la ressource. Dans l'architecture globale, l'identifiant permet au gestionnaire de services de réaliser les différents liens entre les services.
- Le nom permet aux développeurs travaillant sur l'architecture de mieux documenter.
- Le niveau est prédéfini par trois valeurs (critiques, contraint et libre), ce niveau permet de définir la cardinalité de la ressource ; une ressource critique nécessite 1 et seulement 1 lien, une ressource contrainte nécessite au moins 1 lien et une ressource libre ne dispose pas de contrainte sur le nombre de liens.

```
<Ressources>
  <Ressource
    Id = "P1"
    Name = "P1"
    Level = "critique"
    Type = "Provided">
    <Links>
      <Module
        Module = "MO0"
        Port = "0"
        Type = "Ouput"
        Flow = "Event"/>
      <Ressource
        Id = "R1"
        Name = "R1"
        Level = "critique"
        Type = "Provided"
      />
    </Links>
  </Ressource>
  ...
</Ressources>
```

Figure 4-28 : Extrait d'un fichier XML de définition des ressources

La balise <Links> est la « passerelle » avec le middleware ContrAct. Elle permet de faire le lien entre une ressource et les ports de données ContrAct ou de directement créer un lien vers une autre ressource, on parlera alors d'une redirection. Les paramètres de la balise <Module> permettent de définir les éléments suivants (Figure 4-28):

- Le nom du module, dont le port est associé à la ressource.
- Le numéro du port associé au port du module ContrAct concerné.
- Le type peut être de deux valeurs différentes (Input ou Output), ce type correspond au type du port défini dans les modules ContrAct.

- Le paramètre «flow» permet d'indiquer s'il s'agit d'un port de données ou d'un port d'événement de ContrAct, par défaut s'il n'est pas indiqué dans la définition, il sera considéré comme un port de données.

4.9.3 La gestion de service : liaisons basées sur les ports

La Figure 4-29 montre que, comme nous l'avons mentionné, la fonction exploitée par une configuration d'un service est portée par un module ou une composition de modules, et les liaisons sont établies par l'intermédiaire des ports de ces modules.

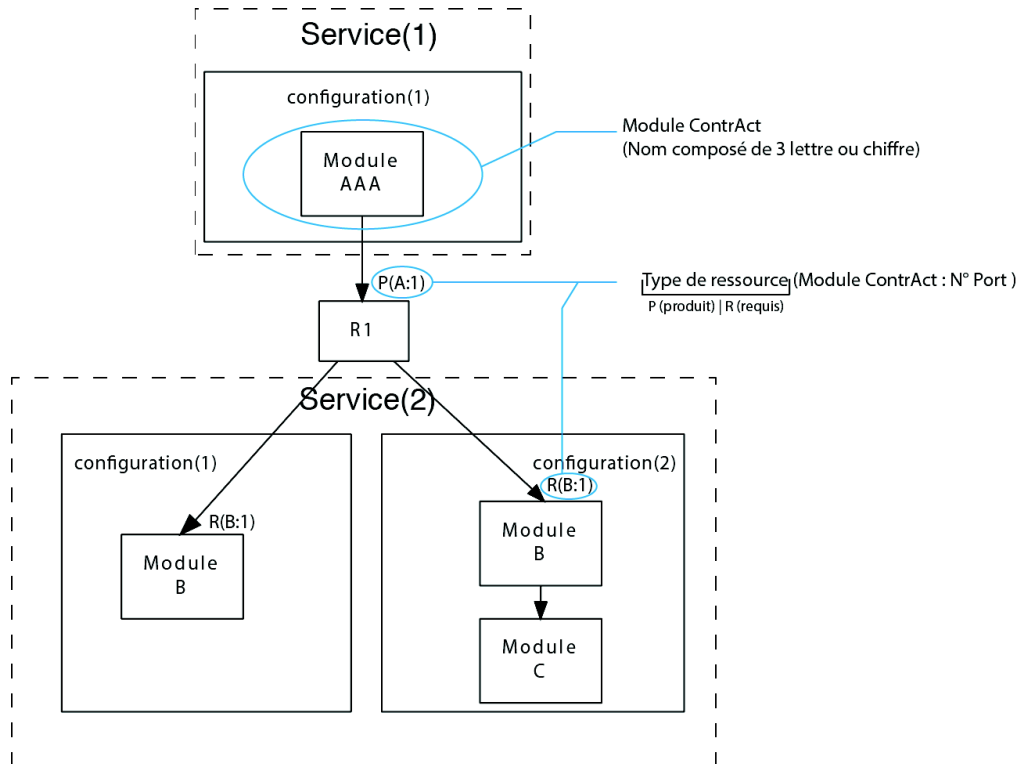


Figure 4-29 : Liaisons basées sur les ports des modules

Le lien entre les services et les modules est basé sur les ports de la façon suivante :

- Port de requête - il correspond au point d'entrée de messages qui permettent à d'autres modules de contrôler son activité. Ce port de requête est utilisé par le gestionnaire de services pour le démarrage et l'arrêt de la fonction, dont l'exécution est sous contrôle de l'ordonnanceur, et l'abonnement des flux entre les services (données et événements). De plus, il est utilisé par les superviseurs de service pour mettre en place les liens entre les modules internes à la fonction.
- Ports de paramètres – ce sont les ports qui permettent la configuration des paramètres des modules, ils sont utilisés par les superviseurs de service si la fonction le nécessite.
- Ports de données (entrées ou sorties) - ils correspondent chacun à un flux rentrant ou sortant du module. Ces ports permettent les interconnexions de données entre modules, et sont assimilés à des ressources ceux qui sont reportés à l'interface des schémas (i.e., de la composition correspondant à la fonction).

- Ports d'événements (d'entrées ou de sorties) - ils correspondent chacun à un flux d'entrée ou de sortie d'événement. Par exemple, on utilise un port d'événement lorsque le superviseur réalise une modification de configuration. Le gestionnaire de services est notifié par un événement afin qu'il réalise les nouvelles interconnexions de flux de données correspondant à la nouvelle configuration.

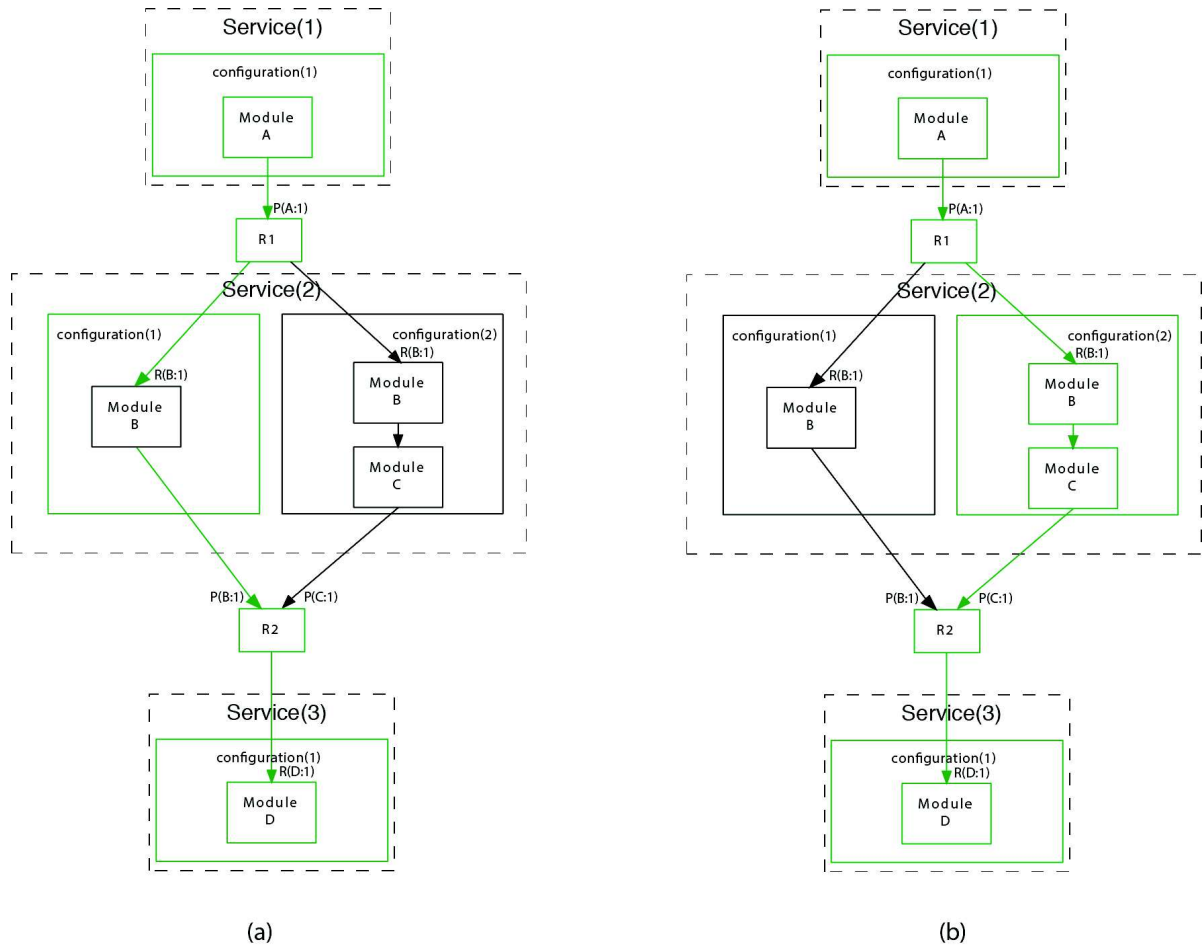


Figure 4-30 : Exemple de liaisons de flux de données

Les liaisons entre les services et les ressources sont illustrées sur la Figure 4-30, en mettant en évidence les relations entre les numéros de port des modules et les ressources. Dans cet exemple, nous avons 3 services, deux avec une configuration unique et le troisième qui dispose de deux configurations. Dans l'exemple (a) et (b), nous allons séquentiellement démarrer les services d'un même mode. Lorsque l'on active le service 1 et que la configuration 1 est active, il n'y a aucun lien de créé ; il faut attendre que le service 2 soit activé et que la configuration soit également sélectionnée afin de créer les liaisons entre le service 1 et 2. Les liaisons de données qui vont être créées dans l'exemple (a) sont les suivantes :

$$\begin{aligned}
 &A: 1 \rightarrow B: 1 \\
 &B: 1 \rightarrow D: 1
 \end{aligned}
 \tag{4.6}$$

Dans (4.6), nous pouvons remarquer deux ports nommés de la même manière (B: 1). Il ne s'agit pas du même port ; dans $A: 1 \rightarrow B: 1$, B: 1 correspond du port d'entrée numéro 1 du module B qui est lié avec le port de sortie numéro 1 du module A alors que dans $B: 1 \rightarrow D: 1$, B: 1 correspond au port de sortie numéro 1 du module B qui est lié avec le port d'entrée numéro 1 du module D. La distinction s'établit par rapport aux ressources. La cardinalité entre les ports est définie dans la déclaration de la ressource, car des ressources consommée (définies par $R(\cdot)$, par exemple $R(B,1)$) ou produites (définies par $P(\cdot)$, par exemple $P(A,1)$) peuvent être aussi bien connectées sur un port de sortie que sur un port d'entrée de flux de données d'un module ContrAct.

Dans l'exemple (b), les liaisons créées sont les suivantes :

$$\begin{aligned} A: 1 &\rightarrow B: 1 \\ C: 1 &\rightarrow D: 1 \end{aligned} \quad (4.7)$$

Notons que, dans l'exemple (b), le lien entre le module B et le module C ($B: 1 \rightarrow C: 1$) ne sont pas décrits. En effet, ces liens sont intrinsèques au service, ou plutôt internes à la fonction associée au service. Dès lors, ils ne sont pas décrits lors de la définition du service, i.e. ils ne concernent pas les liens entre services et ressources. Ces liens sont en fait définis dans le superviseur associé au service, qui lui met en place la fonction (i.e., le schéma de modules correspondant à la fonction). Donc, seuls les liens mettant en œuvre des ressources « externes » sont décrites dans la description de service (à travers les fichiers XML présentés).

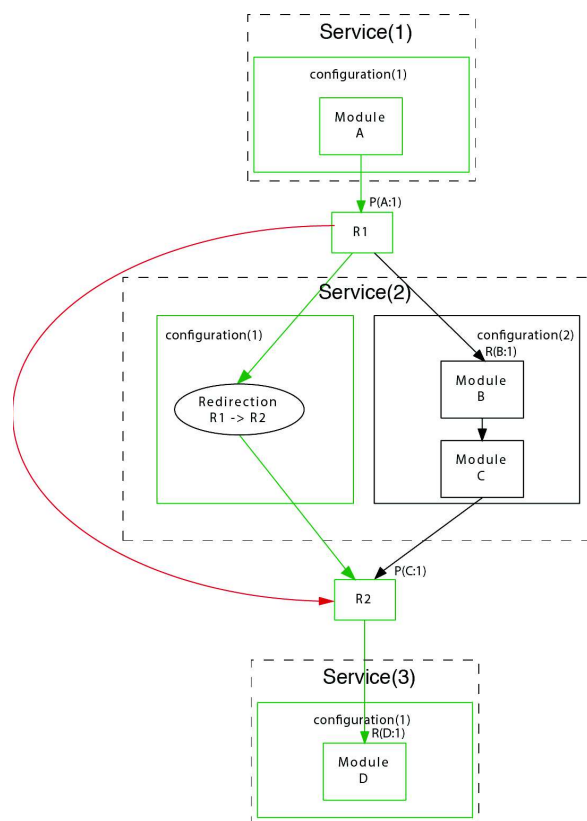


Figure 4-31 : Exemple d'une redirection d'un flux de données

Dans la Figure 4-31, nous décrivons un autre type de liaison, correspondant à une redirection de ressource. Dans la description d'un service, une ressource peut être définie comme une redirection lorsque la configuration du service ne contient pas de fonction associée et établit seulement le lien entre une ressource entrante (requis, ou consommée) et une ressource de sortie (produite). C'est le cas sur la figure pour la configuration 1 du service 2 (lien en rouge sur la Figure 4-31). Quand cette configuration est sélectionnée, les liens suivants sont créés :

$$A: 1 \rightarrow D: 1 \quad (4.8)$$

Une ressource peut être requise (consommée) par plusieurs configurations de plusieurs services, et donc par plusieurs modules (ou par plusieurs redirections). La validité des liens en termes de nombre de liaisons est vérifiée selon le niveau de la ressource : critique, contrainte ou libre. Ce niveau permet d'affecter, et donc de vérifier, un nombre minimal et/ou maximal de liaisons avec une ressource :

- Une ressource critique doit nécessairement avoir une et seulement une liaison,
- Une ressource contrainte doit obligatoirement avoir une liaison
- Une ressource libre, ne dispose d'aucune limitation.

Cette validation est effectuée par le gestionnaire de service lorsqu'il vérifie la cohérence globale de l'architecture lors du chargement en mémoire du graphe complet de l'architecture. La Figure 4-32 est un exemple de graphe pouvant être traité par le système. Celui-ci représente l'ensemble des relations entre les services. Ces relations entre les services sont réalisées par des liaisons entre les ressources produites par l'un et celles consommées par l'autre.

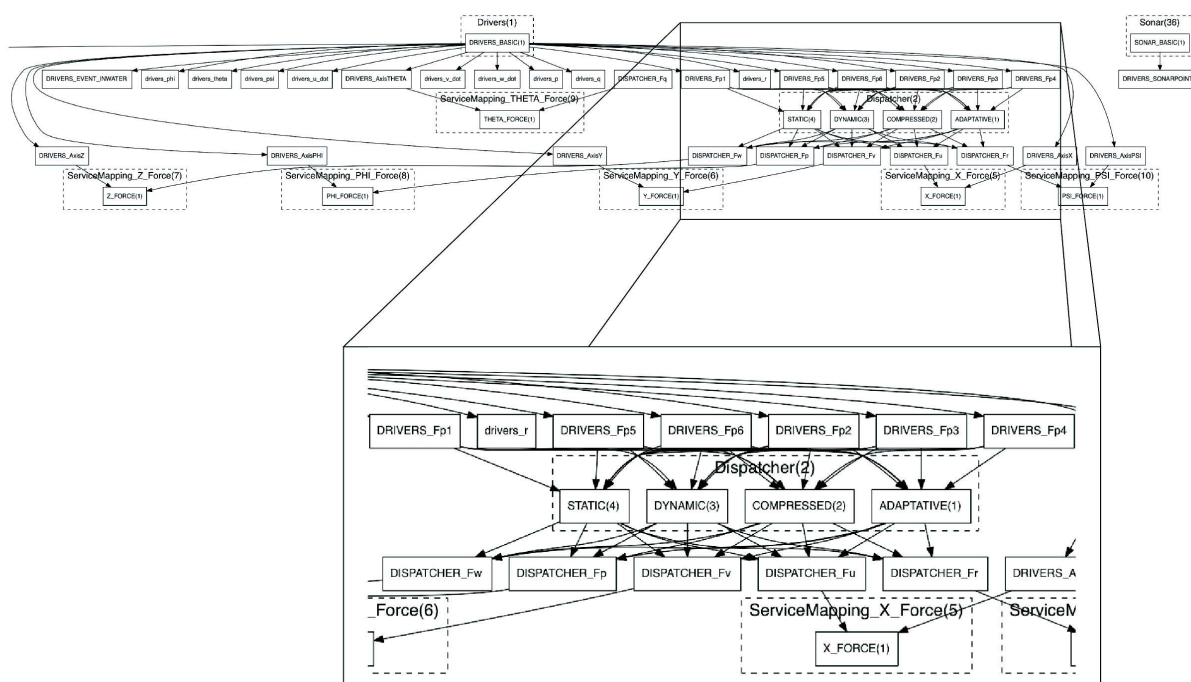


Figure 4-32 : Exemple de graphe d'une architecture globale

Nous pouvons constater que la complexité du graphe de l'architecture globale peut être d'un niveau très élevé et qu'il y a une multitude de combinaisons possibles entre les services. Les différentes combinaisons sont définies dans la définition des modes. Un mode, comme défini précédemment, est une composition de services permettant d'apporter une fonction (agrégée) à l'opérateur par exemple un asservissement en cap ou en profondeur.

4.9.4 La gestion de service : changement de mode

Nous avons considéré que le système dispose d'un mode par défaut (Figure 4-33) et que lorsqu'un changement de mode est réalisé, seuls les services non communs entre le mode par défaut et le nouveau mode sont remplacés.

Le mode par défaut est représenté sur la Figure 4-33. Ce mode par défaut est le contrôle du robot le plus basique, il permet de contrôler celui-ci en appliquant les forces désirées directement aux entrées du répartiteur que nous avons préalablement décrit dans le chapitre 3.

La lecture de cette figure se fait de gauche à droite, indiquant le sens de production des ressources ; les ressources consommées se trouvent sur la gauche et les produites sur la droite du service. Chacune des « boîtes » correspond à un service qui produit et consomme des ressources. Le service « Drivers » le plus à gauche est le service le plus proche de la partie matérielle. Cette partie matérielle ne requiert pas de ressource et ne sera pas modifiée lors d'une mission, car la configuration matérielle ne changera pas (par exemple, pas d'ajout d'un capteur au cours de la mission).

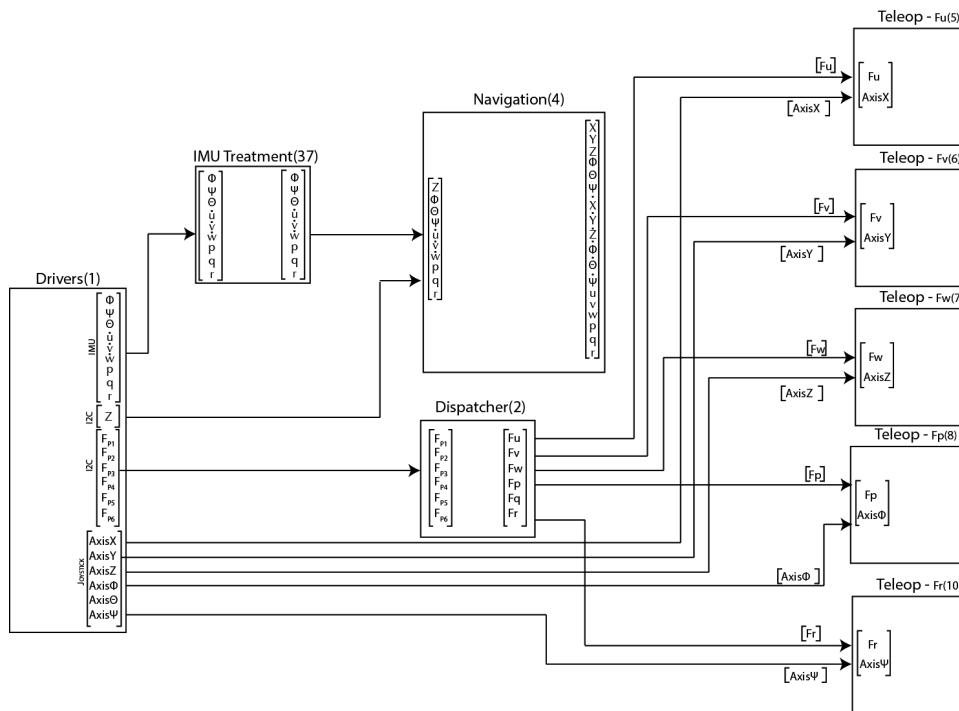


Figure 4-33 : Mode de commande en force (mode par défaut)

Un service est symboliquement représenté sur la Figure 4-34, avec les modules contenus dans les diverses configurations. Cette représentation permet également de visualiser les ressources produites et requises. Les ressources produites sont toujours identiques quelle que soit la configuration.

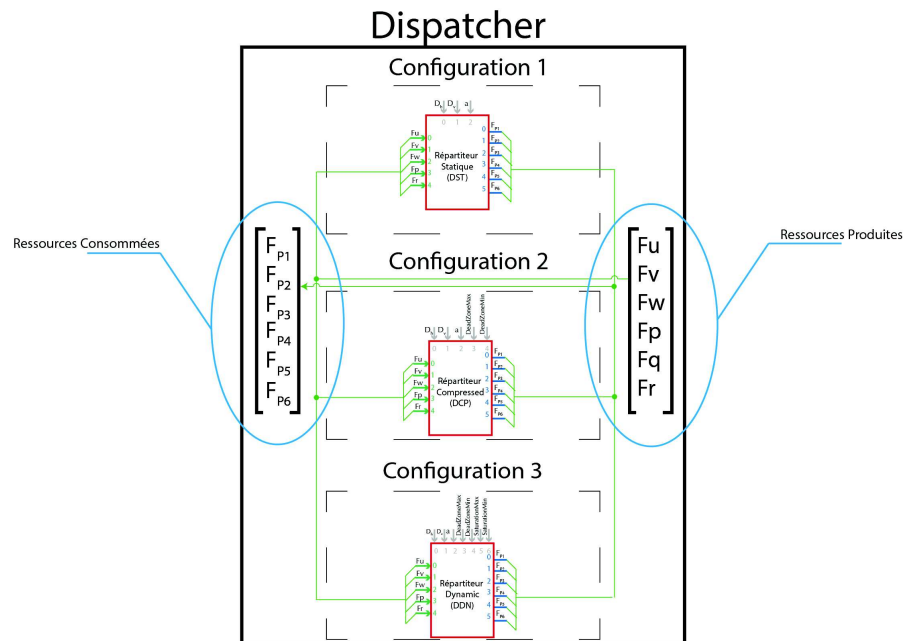


Figure 4-34: Représentation symbolique d'un service

Dans le cas de l'activation du mode permettant l'asservissement en cap (Figure 4-35) ou d'un asservissement du cap et de la profondeur (Figure 4-36), l'algorithme qui permet le changement de mode va déterminer quels sont les services non communs entre l'ancien mode et le nouveau.

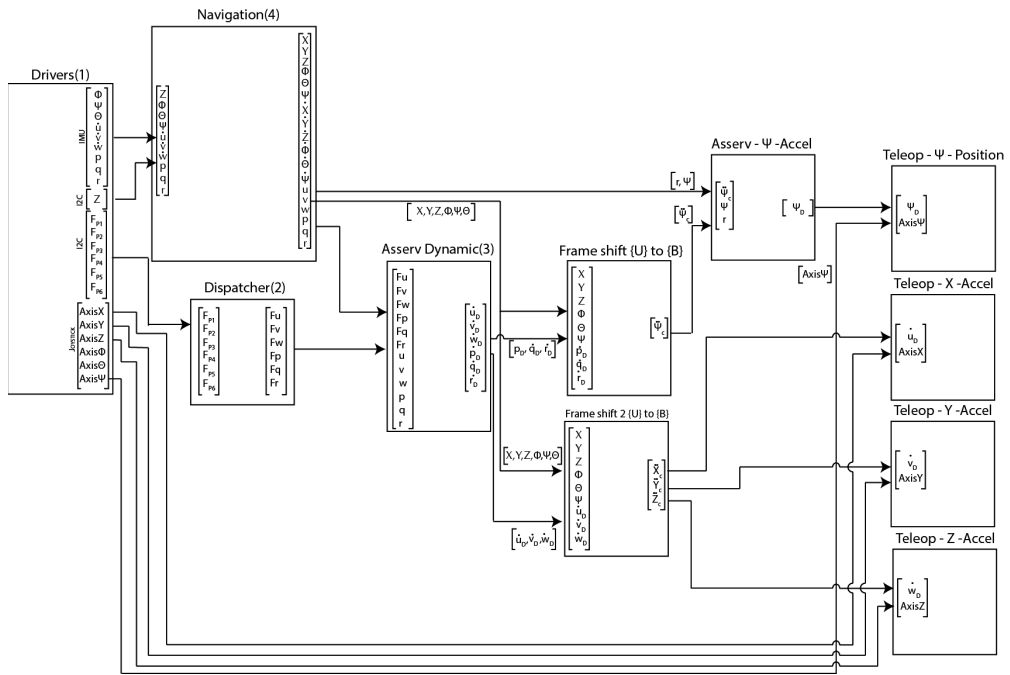


Figure 4-35 : Mode de fonctionnement : asservissement en cap

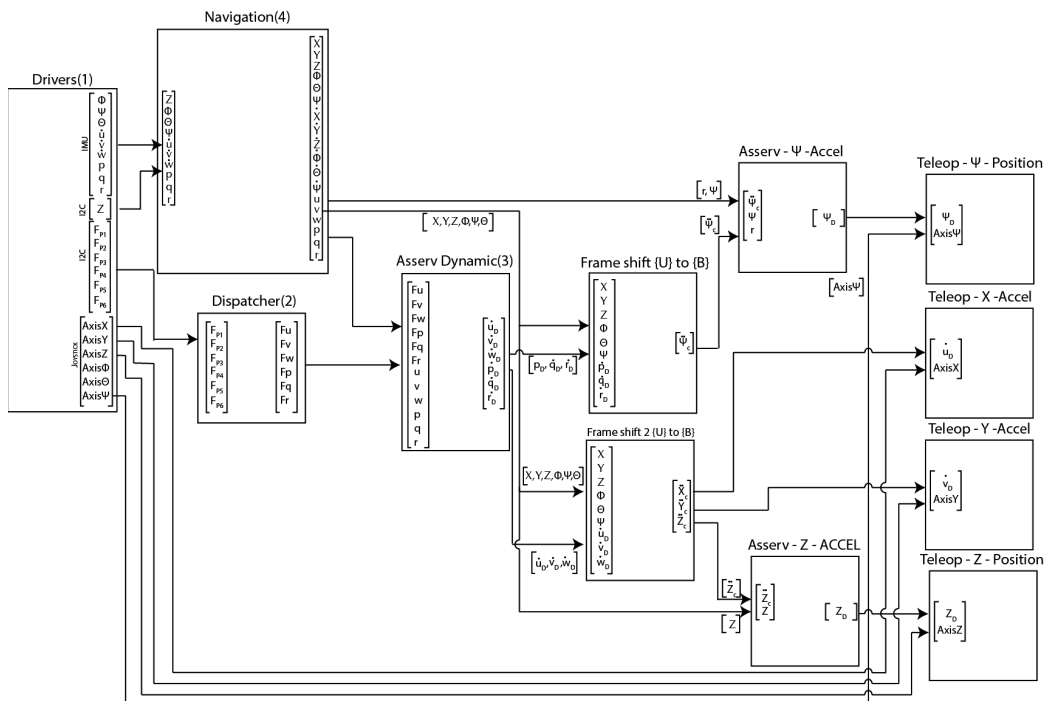


Figure 4-36 : Mode de fonctionnement : asservissement en cap et en profondeur

Le changement de mode est illustré sur la Figure 4-37: le (a) en vert est le service par défaut (ex. Figure 4-33) et le (b) en rouge est un service plus évolué (ex. Figure 4-36). Les services communs entre les deux modes sont entourés en bleu. Prenons l'exemple où le système est dans le mode par défaut (a) et que l'on souhaite passer dans le mode plus évolué (b). L'algorithme fait la différence entre le mode (a) et le mode (b), cette différence est réalisée de la façon suivante :

$$(a) \oplus (b) \tag{4.9}$$

où le résultat de cette différence est l'ensemble des services qui doivent être activés ou désactivés lors de l'action du nouveau mode (Figure 4-38(1)). Les services à désactiver sont déterminés dans la relation (4.10) et illustrés dans la Figure 4-38(2).

$$(a) \cdot ((a) \oplus (b)) \tag{4.10}$$

Les services à activer sont déterminés par la relation suivante et illustrés dans la Figure 4-38(3).

$$(b) \cdot ((a) \oplus (b)) \tag{4.11}$$

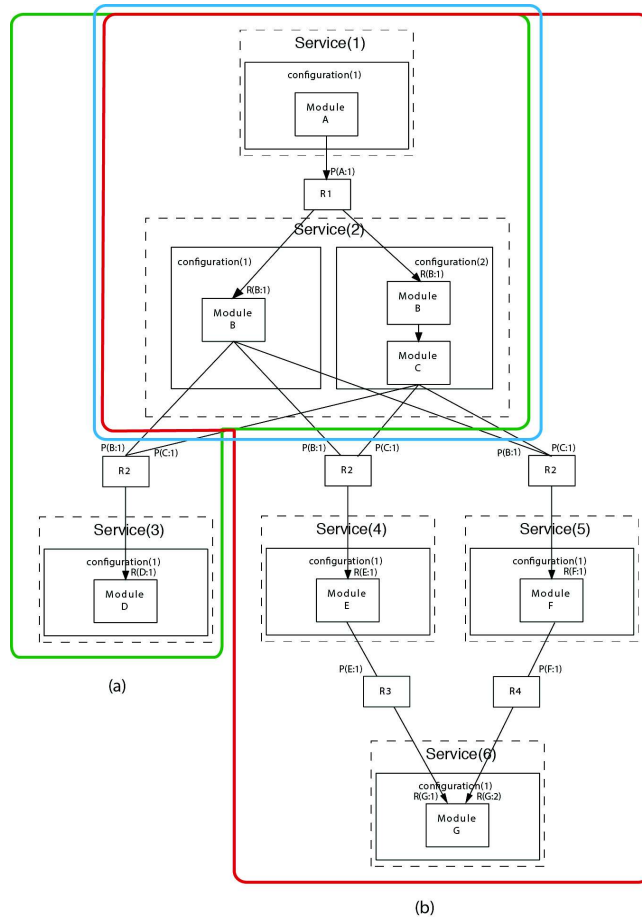


Figure 4-37 : Les changements de modes

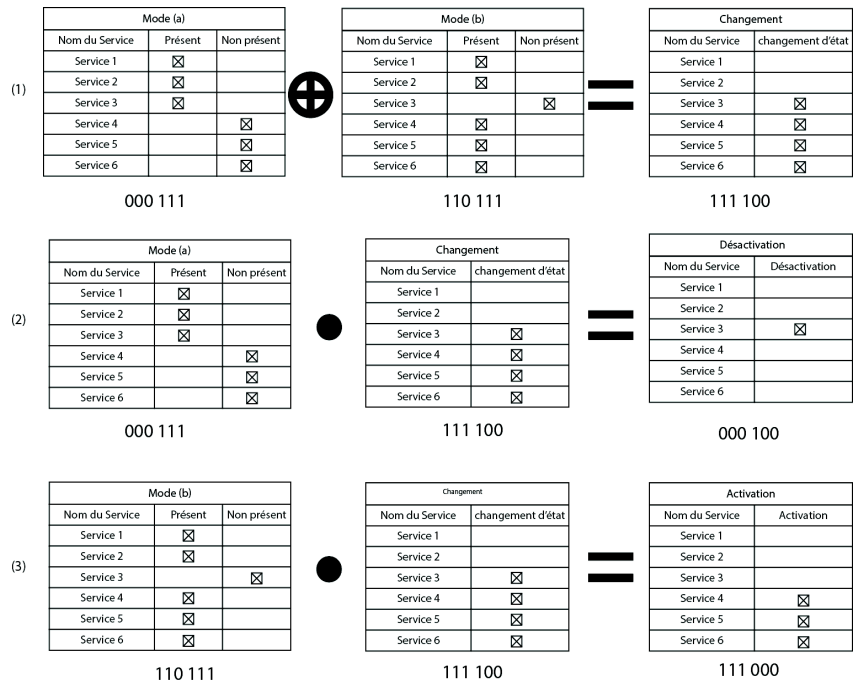


Figure 4-38 : Opérations réalisées lors d'un changement de services

Donc pour passer du mode par défaut (Figure 4-33) au mode d'asservissement du cap (Figure 4-35), ou vice-versa, on utilisera cet algorithme de différenciation. La Figure 4-39 permet de visualiser les différents changements de mode réalisés, par l'opérateur ou par un contrôle de plus haut niveau, lors de la mission. Sur cette figure, 3 changements de modes sont retracés : en bleu le mode initial (mode par défaut, au démarrage), en rouge le mode permettant de réaliser un asservissement en cap et le mode en vert est un asservissement du robot en cap et en profondeur (Figure 4-36). Chaque ligne indique l'implication des modules dans ces services.

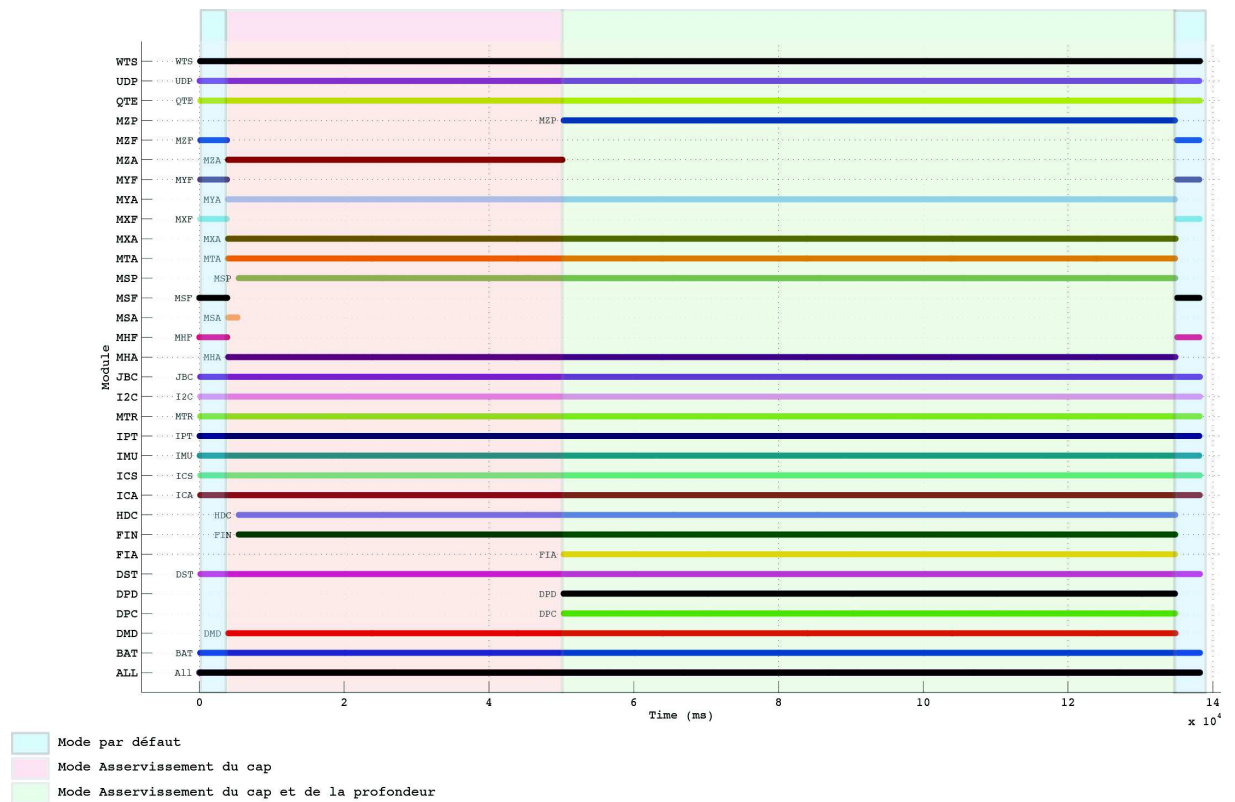


Figure 4-39 : Visualisation de changements de modes

4.9.5 La gestion de service : exécution sous contrôle de l'ordonnanceur

Lorsqu'un changement de mode est réalisé, l'ordonnanceur réorganise le séquençage des modules. Ce séquençage est réalisé de façon périodique, dans notre cas la période est de 100 ms. La Figure 4-40 représente l'activité de tous les modules²⁵ à chaque période de 100ms. La représentation adoptée sur la Figure 4-41 est la suivante : à chaque module est associé un code couleur (le même que celui utilisé sur la Figure 4-39) et chaque début (respectivement fin) d'exécution est indiquée par un rond (resp. une croix).

Nous constatons que les contraintes temporelles du contrôle sont respectées.

²⁵ Tous les modules sont répertoriés dans l'Annexe I

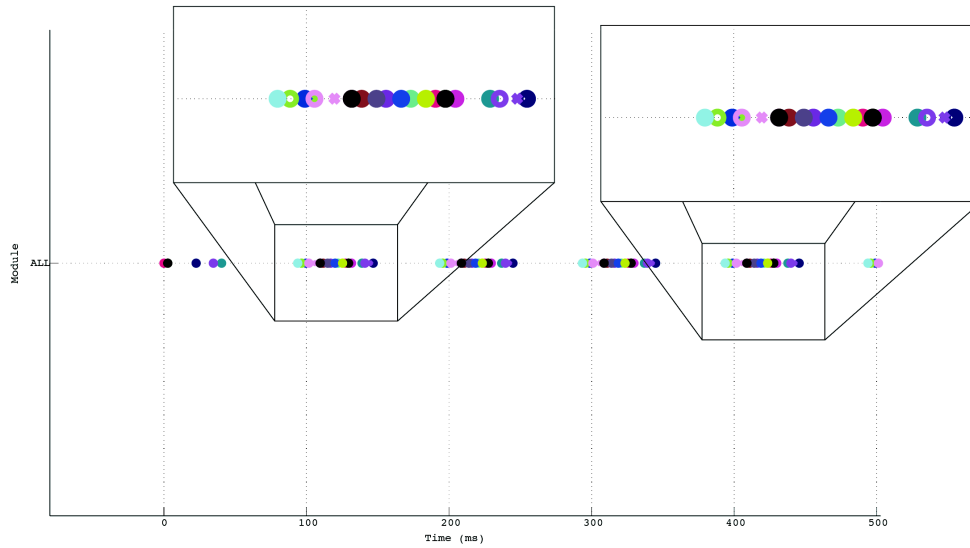


Figure 4-40 : Ordonnancement périodique

Les durées d'exécution (maximum, minimum, moyenne ainsi que l'écart type) des différents modules utilisés dans l'architecture²⁶ sont indiquées dans le Tableau 4-1 et représentées sur la Figure 4-41. Nous pouvons constater que le temps cumulé de cet ensemble de modules est de 10.79 ms, avec une faible variation (écart type d'environ 1% sur la période). Il faut préciser que ces durées ne prennent pas en compte les threads associées aux modules concernés (les threads ayant une priorité supérieure). En effet, pour les modules dédiés aux capteurs les opérations de lecture et écriture (fournies par les drivers) sont déportées dans des threads afin de minimiser l'impact de ces opérations sur l'ordonnancement car elles dépendent directement des capteurs (par exemple, le sonar profilométrique envoie lui-même ses données lorsqu'elles sont disponibles). Donc le temps d'utilisation processeur sur une période est supérieur à 10.79 ms (pour une période de 100 ms).

²⁶ Les modules sont décrits dans l'Annexe I

Module	Durées (ms)			
	Maximum	Minimum	Moyenne	Ecart type
DMD	1.26	1.14	1.17	0.008
DPC	0.35	0.29	0.30	0.006
DPD	0.28	0.16	0.17	0.005
DST	0.43	0.40	0.41	0.006
FAC	0.53	0.31	0.32	0.007
FIA	0.49	0.32	0.33	0.007
FIN	0.47	0.31	0.32	0.006
ICA	0.36	0.34	0.34	0.006
ICS	0.56	0.40	0.42	0.007
IMU	0.12	0.10	0.11	0.003
IPT	0.24	0.11	0.16	0.004
MTR	1.49	1.00	1.03	0.017
I2C	3.59	3.18	3.28	0.046
JBC	0.41	0.38	0.39	0.007
MHA	0.06	0.04	0.04	0.002
MSP	0.07	0.05	0.05	0.002
MTA	0.06	0.04	0.04	0.002
MXA	0.06	0.04	0.05	0.002
MYA	0.06	0.04	0.04	0.002
MZP	0.07	0.05	0.05	0.002
QTE	0.19	0.15	0.18	0.004
UDP	1.27	0.28	0.82	0.095
YWC	0.33	0.29	0.30	0.006
TOTAL	10.79	7.10	10.32	0.128

Tableau 4-1 : Durées d'exécution des modules

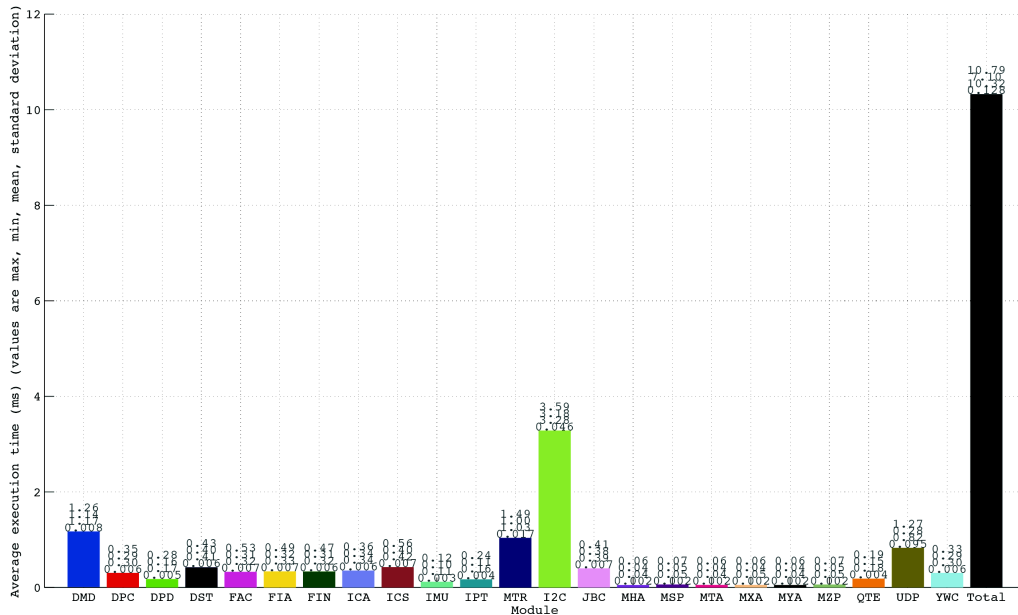


Figure 4-41 : Durée d'exécution des modules

Pour chacun des modes de fonctionnement (Figure 4-42), nous observons des ordonnancements différents, et chaque ordonnancement est logiquement différent d'un mode à l'autre puisque ce ne sont pas les mêmes fonctions (i.e., les mêmes séquences de modules impliqués).

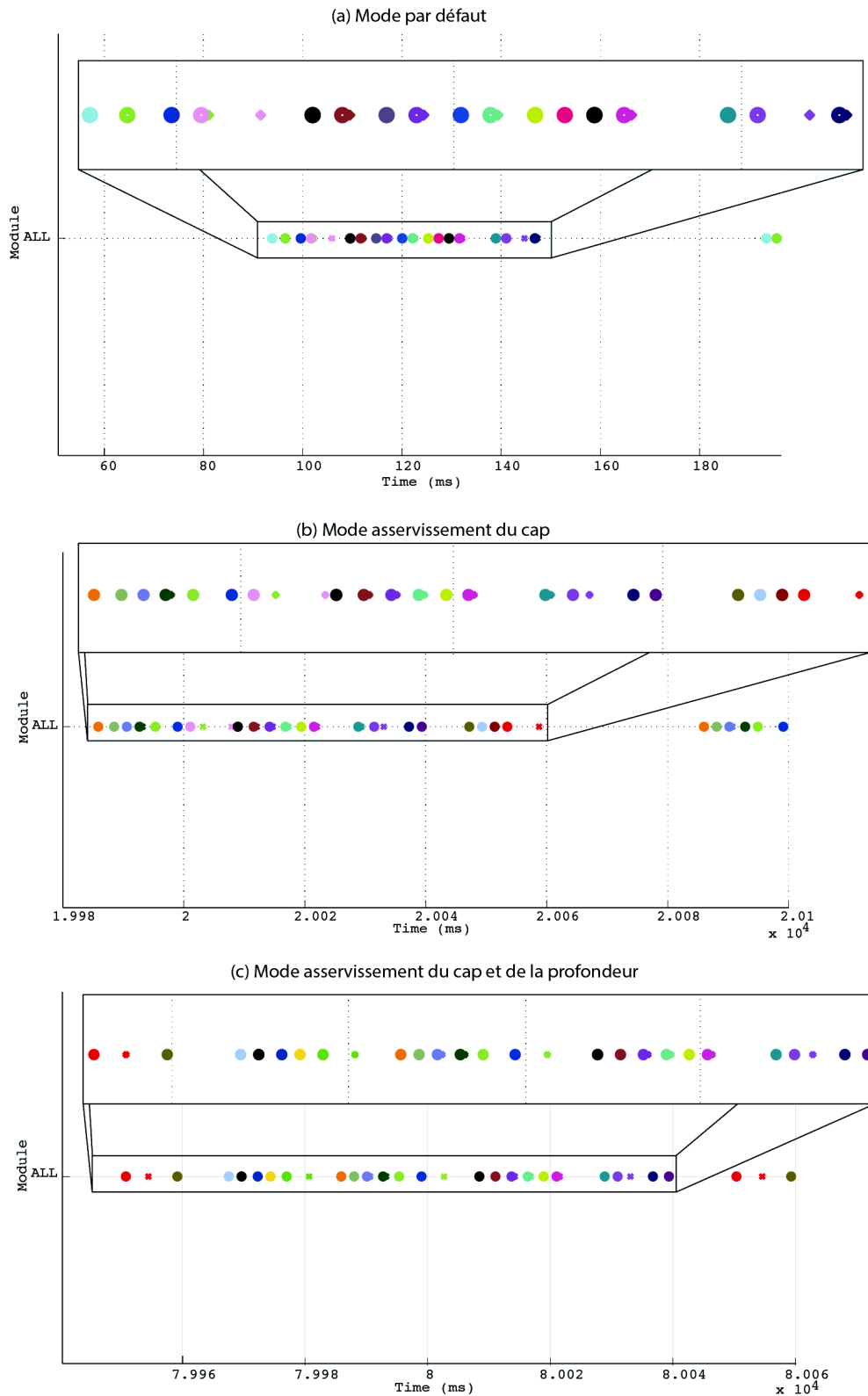


Figure 4-42 : Ordonnements des modules selon les modes

Néanmoins, pour chaque mode l'ordonnancement devrait toujours être le même ; or ce n'est pas ce que nous observons. Ce comportement, non désiré, est dû à la non expression des contraintes de précédences entre les schémas ; en effet, la version actuelle de ContrAct ne permet pas d'exprimer des contraintes de précedence entre schémas, c'est à dire entre modules relevant de schémas différents. Il s'agit d'une limitation importante car les relations de précedence définissent l'ordre précis dans lequel les calculs doivent effectués dans le cas d'expressions complexes (i.e., impliquant plusieurs modules, relevant potentiellement de schémas différents). Le middleware ContrAct ne permettant que l'expression de contraintes de précedence au sein d'un même schéma, les calculs peuvent ne pas être exécutés dans le « bon ordre », induisant des retards, voire même des erreurs, dans le contrôle du robot.

Dans la section suivante, nous allons justement nous intéresser aux problèmes liés au défaut d'expression de contraintes de précedence, et nous allons constater directement son impact sur le comportement du contrôle.

4.9.6 Impact des contraintes de précedence non respectées

Considérons l'exemple basique décrit sur la Figure 4-43. Lorsque l'ordonnancement respecte les contraintes de précedence, il n'y a pas de retard ni dans la production ni dans l'acquisition des données par les modules concernés (Tableau 4-2). Par contre dans le cas du non-respect de ces contraintes (Tableau 4-3), il est possible d'atteindre, dans le cas le moins favorable, un retard de 300 ms, déterminé selon l'expression suivante :

$$Retard = Nombre\ de\ module * Periode \tag{4.12}$$

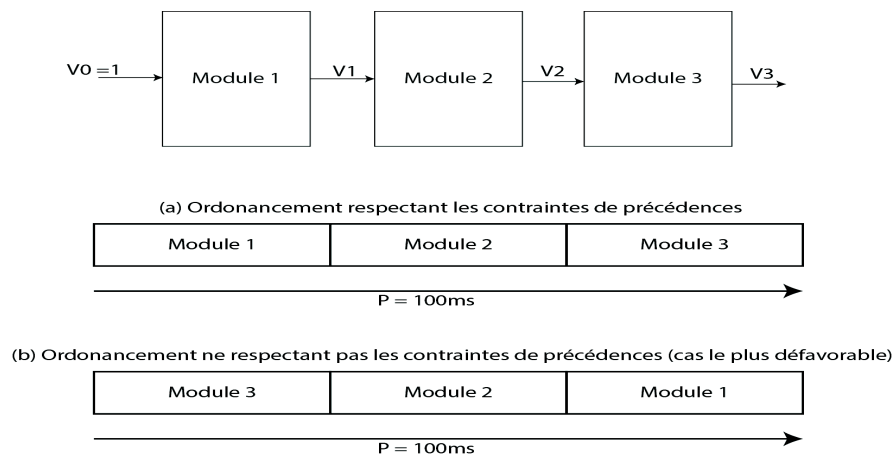


Figure 4-43 : Exemple de non respect de la précedence des modules

	Module 3	Module 2	Module 1
<i>cycle_i</i>	V3 = 1	V2 = 1	V1 = 1

Tableau 4-2 : Résultat dans le cas du respect des contraintes de précedence

	Module 3	Module 2	Module 1
$cycle_i$	V3 = 0	V2 = 0	V1 = 1
$cycle_{i+1}$	V3 = 0	V2 = 1	V1 = 1
$cycle_{i+2}$	V3 = 1	V2 = 1	V1 = 1

Tableau 4-3 : Résultat dans le cas du non respect des contraintes de précédence

Nous avons également observé ce phénomène dans le cadre d'expérimentations avec la plateforme dans une configuration nominale. La Figure 4-44 permet visualiser l'impact des différents retards qui sont liés au non respect des contraintes de précédence lors d'un mode permettant l'asservissement du cap. La courbe bleue du premier graphique (haut) est le comportement que devrait réaliser la plateforme si les précédences étaient respectées et la courbe rouge est le comportement réel du robot. Nous pouvons constater l'erreur induite : même si elle apparaît faible sur le graphique, elle peut atteindre 6 %.

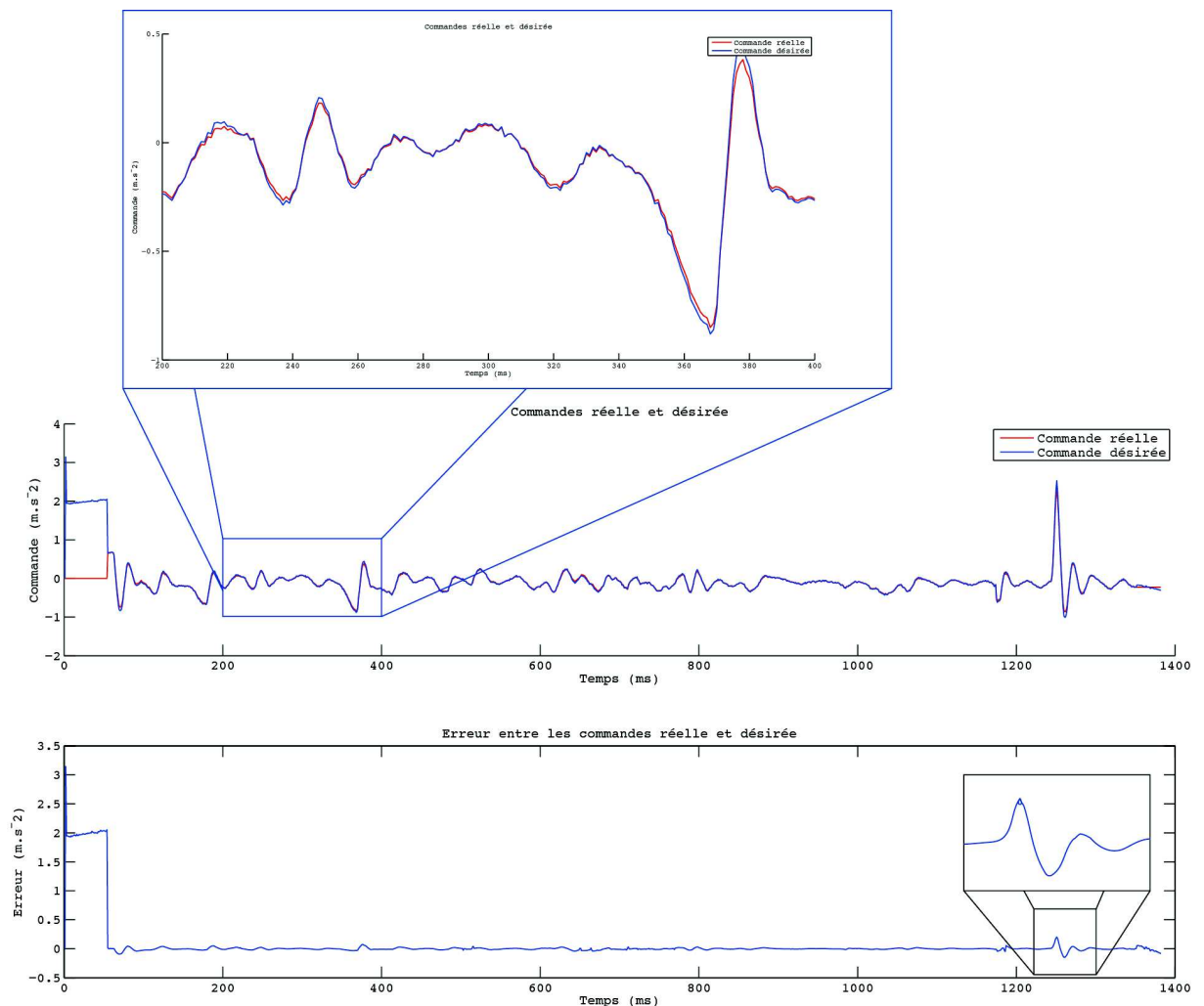


Figure 4-44 : Visualisation d'une commande dont l'ordonnancement ne respecte pas les précédences

Pour pouvoir supprimer ce problème de retard, tout en continuant d'utiliser le middleware ContrAct dans sa version actuelle, nous avons supprimé la notion de multi-schéma dans une même architecture. En fait, nous avons supprimé le fait qu'un service était constitué d'un ensemble de configurations, où la configuration était directement assimilée à un schéma, et nous avons décrit un seul (et unique) schéma pour tout le mode (i.e., un schéma impliquant tous les modules de tous les services impliqués dans le mode). Chaque mode a bien sûr son schéma global.

La Figure 4-45 est la nouvelle représentation, compensant le problème, de notre structuration avec ContrAct. Le terme schéma correspond, dans cette version « adaptée » (et provisoire), à un mode de fonctionnement (en violet), donc à un ensemble de services composés et non plus à un service comme dans la Figure 4-20.

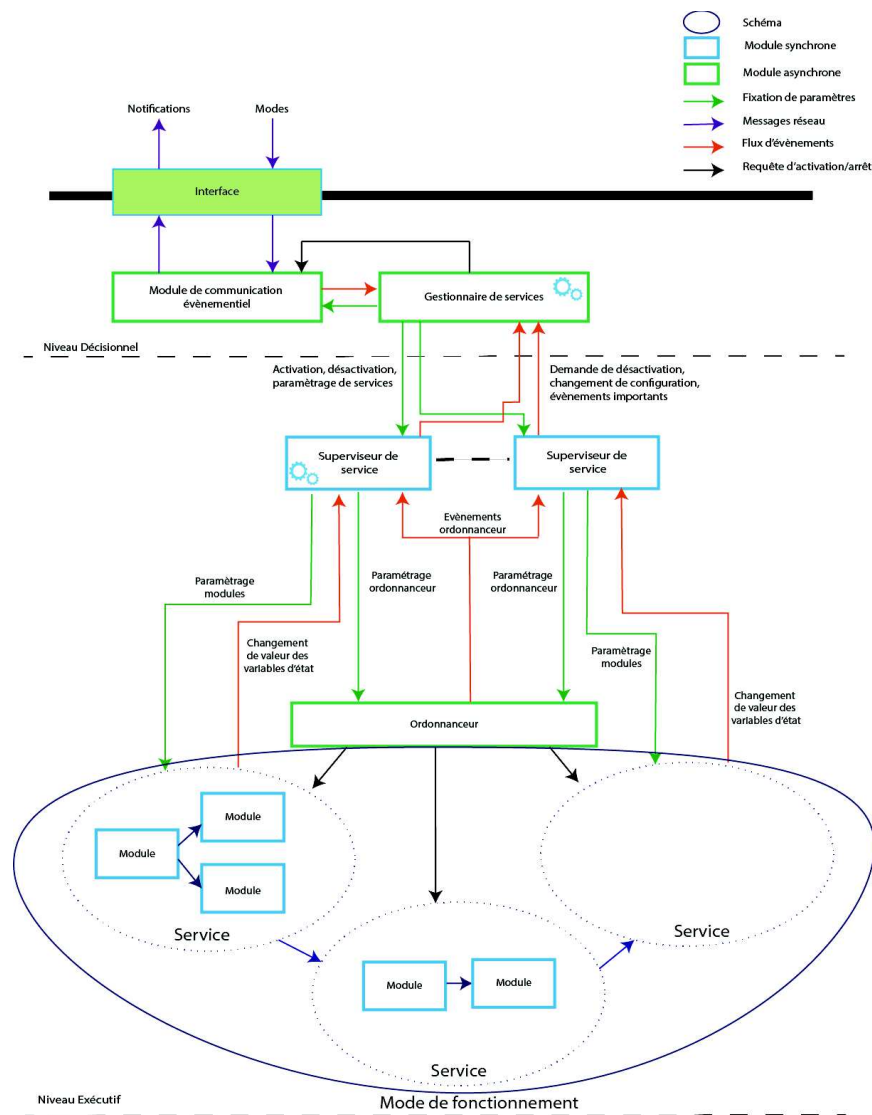


Figure 4-45 : Nouvelle structuration de l'architecture basée sur un schéma global par mode

L'ordonnancement du mode d'asservissement du cap est donné sur la Figure 4-46, mais cette fois-ci le mode est défini dans un seul schéma ContrAct. L'utilisation d'un schéma unique permet à l'ordonnanceur de respecter les précédences exprimées dans les superviseurs ContrAct. Nous pouvons constater que l'ordonnancement est complètement différent de celui de la Figure 4-42(b), les deux étant indiqués sur la Figure 4-46 pour comparaison visuelle. Cet ordonnancement sera identique à chaque activation de ce mode de fonctionnement.

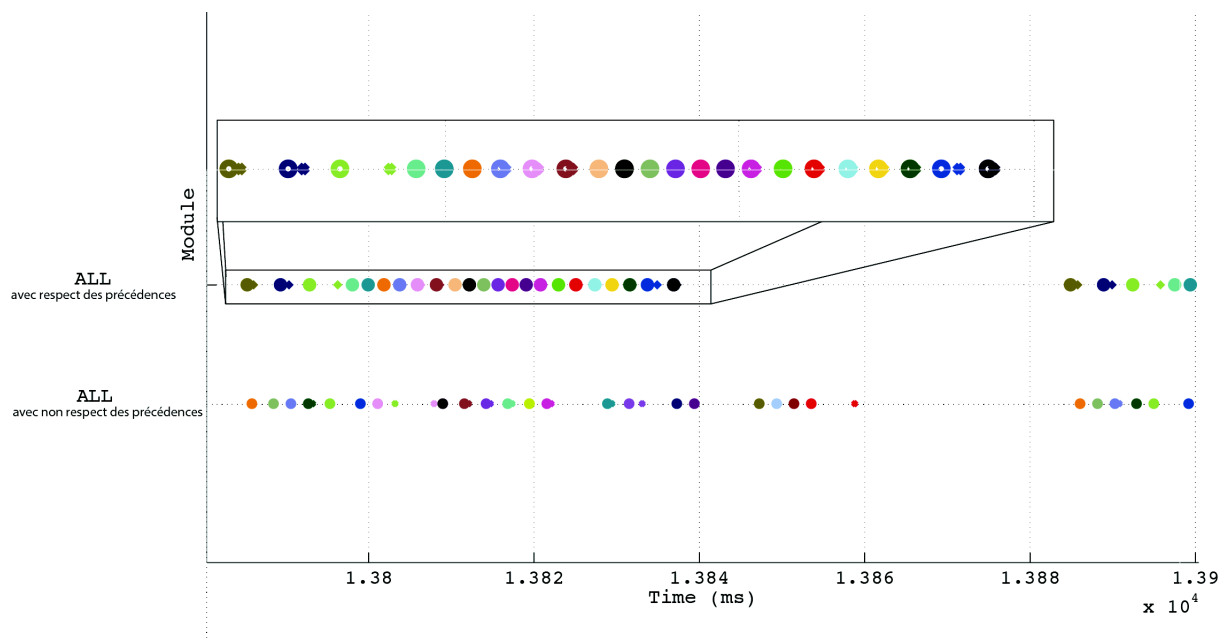


Figure 4-46 : Ordonnancement d'un mode défini statiquement dans un même schéma ContrAct

Comme nous pouvons le constater à travers la Figure 4-47, lorsque l'ordonnancement respecte les précédences, les commandes réelles appliquées par le robot sont identiques aux commandes désirées. L'erreur est nulle durant toute l'activité de ce mode, ce qui n'était pas le cas lorsque l'ordonnancement ne respectait pas les précédences (Figure 4-44).

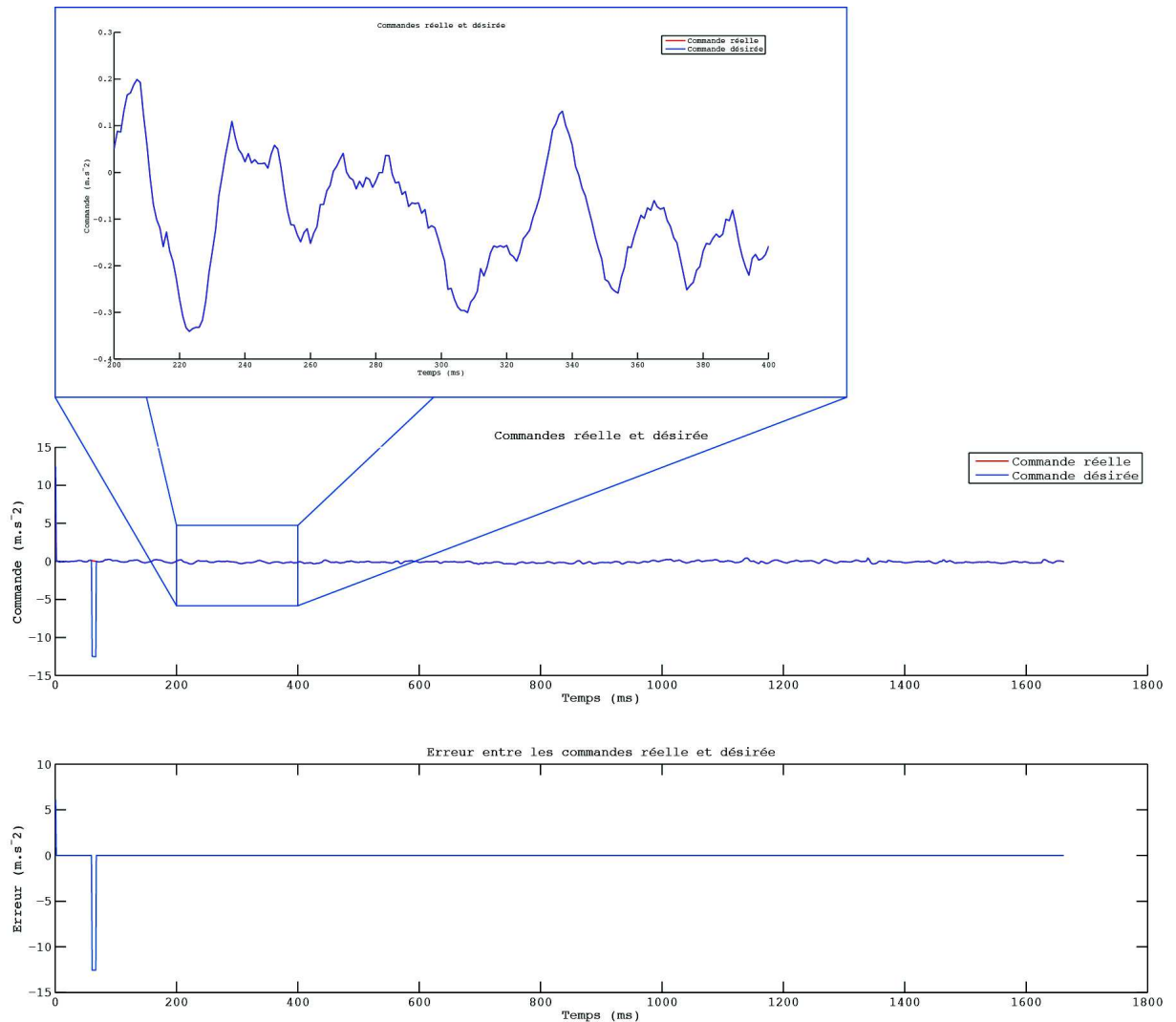


Figure 4-47 : Visualisation d'une commande dont l'ordonnancement respecte les précédences.

Par contre, un autre phénomène que nous devons considérer est la performance temporelle de la commutation de mode. A ce stade, en se basant sur un schéma global (pour compenser le problème de précédence), le temps de désactivation d'un mode et d'activation du suivant est très pénalisé par cette solution. Nous pouvons le constater sur la Figure 4-48 où la commutation du mode par défaut vers le mode asservissement en cap induit un temps de commutation de 3.41s. Ce temps est lié au nombre de liens qu'il faut supprimer et recréer lors du changement mode (globalité des liens), il prend également en compte l'arrêt et le démarrage des modules et le calcul de l'ordonnancement pour le nouveau mode.

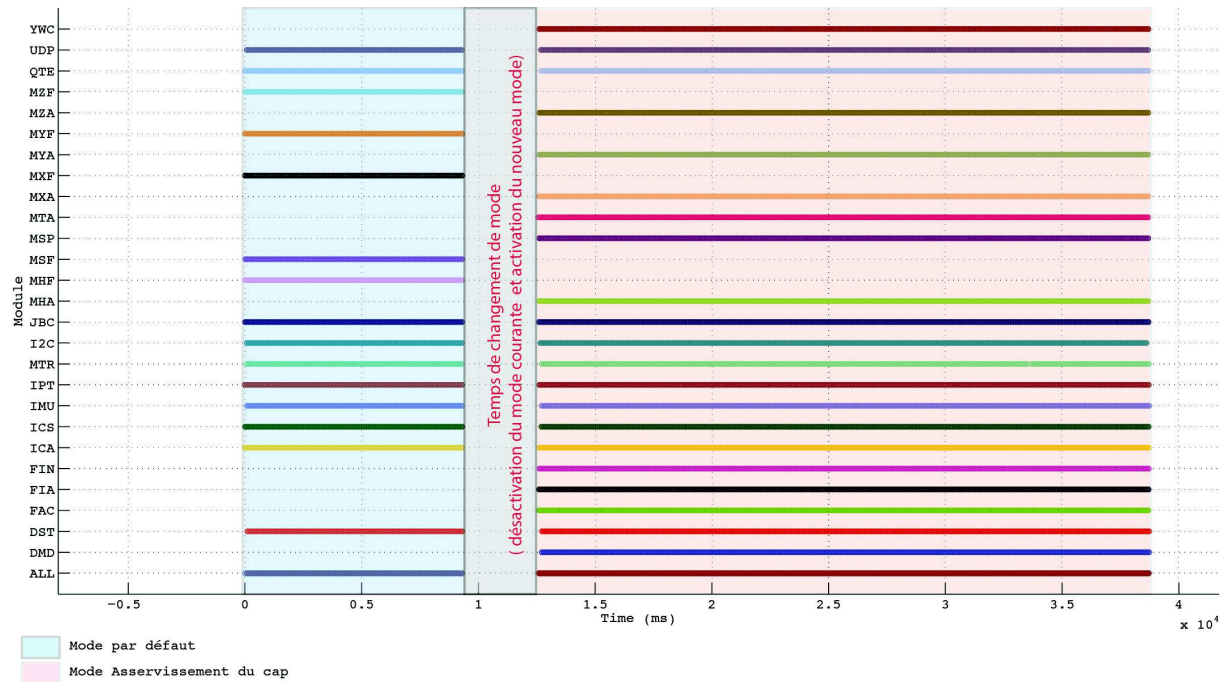


Figure 4-48 : Commutation de mode dans le cas d'un schéma global

4.10 Points clefs

- L'approche orientée service et entités de base composées (modules) permet une structuration « ouverte » de l'architecture ; une modularité nécessaire à la polyvalence du vecteur, favorisant l'évolution et la réutilisation d'entités de base pour construire différents services selon leur composition.
- Cette entité que l'on nomme *Service* est l'entité essentielle de l'architecture. A l'articulation entre les *Modes de fonctionnement* et les entités de base, elle définit l'abstraction de la capacité du robot à réaliser des fonctions robotiques tout en mobilisant des *Ressources* selon des *Configurations*. Ces configurations qui exploitent/créent des ressources disposent de *Conditions d'activation* et d'un *domaine de validité* afin de sélectionner la bonne configuration suivant le contexte dans lequel le robot se trouve.
- Le concept de *Ressource* est réifié afin d'assurer une composition cohérente des entités de base, au sens des contraintes à respecter pour avoir une composition valide. Différents types de ressources sont considérés.
- Cette architecture a été projetée sur le middleware temps-réel ContrAct. Deux entités ont été introduites, selon le modèle ContrAct (i.e., modules), l'une pour gérer les services et l'autre pour superviser un service (i.e., un superviseur par service).
- Un certain nombre de tests ont été réalisés pour valider le comportement du robot ainsi que les contraintes liées à l'ordonnancement lors de l'exécution des services.

Une limitation a été identifiée dans la projection des compositions sur le middleware ContrAct. En effet, chaque composition de modules (service) est implémentée sous la forme d'un schéma, or il n'est pas possible, dans cette version de ContrAct, d'ordonner les schémas. De fait, les contraintes de précédence au sein d'une composition sont vérifiées mais pas celles entre les compositions. Par conséquent, nous avons modifié notre règle de projection pour nous adapter à la situation et ainsi rendre exploitable notre architecture avec cette version du middleware.

Chapitre 5 : Expérimentations

De nombreuses expérimentations ont été réalisées dans le cadre de ces travaux, certaines en piscine, d'autres sur le terrain ; beaucoup de ces données ont été d'ailleurs utilisées à travers le manuscrit pour illustrer nos propositions des chapitres 3 et 4. Nous allons brièvement préciser les principales expérimentations effectuées.

Ces expérimentations ont permis la prise en main de la technologie et son évolution, puisque plusieurs prototypes ont été concrétisés à travers cette thèse. Les expérimentations en piscine ont permis notamment la prise en main des différents capteurs embarqués et les mises au point des différents contrôles nécessaires pour mener à bien les expérimentations prévues sur le terrain. La faible logistique nécessaire pour mettre en œuvre le robot fut un réel avantage dans les différentes campagnes de test, car l'ensemble du matériel nécessaire pour l'utilisation du robot tient dans le coffre d'une voiture (Figure 5-1). Dans cette Figure 5-1, nous pouvons constater que nous respectons la contrainte de faible encombrement qui nous était imposée. La question de la logistique disponible pour les expérimentations est primordiale. Les expérimentations à venir, avec nos collègues hydrogéologues et biologistes marins, vont occasionner une charge logistique accrue. Nous revenons sur le programme expérimental qui poursuivra ces travaux dans le chapitre suivant. Notons toutefois que les premières expérimentations dans le karst (Gourneyras, Source du Lez) devront être conduites en compagnie d'une équipe de spéléo-plongeurs et des partenaires hydrogéologues, ce qui va occasionner un travail de coordination important. De plus, suivant le succès des premières opérations, les phases expérimentales suivantes alourdiront encore la logistique. La maîtrise de nos capacités expérimentales est une question cruciale qu'il ne faut pas négliger.



Figure 5-1 : Le jack et son extension d'actionnement

5.1 Les Expérimentations en piscines

Ces expérimentations en piscines (Figure 5-2) ont permis, entre autres, de mettre en avant les problématiques exprimées dans le chapitre 3 et 4. Car dans la théorie et en simulation, on prend faiblement en considération les problèmes liés aux perturbations et imperfections du « terrain », au sens large, notamment la disparité d'actionnement. Ce sont d'ailleurs ces expérimentations qui ont mis en évidence deux points essentiels :

- L'ampleur de la disparité des caractéristiques des moteurs. En effet, nos moteurs ont (bien sûr) des caractéristiques différentes, ce qui occasionne un couplage entre les degrés de liberté (DDL) dans la relation entre les efforts désirés F_B^d (issus du contrôle ou de l'opérateur) et ceux effectivement induits par l'action des moteurs F_B . Nous avons tout d'abord procédé par 'tâtonnements' pour comprendre le phénomène, et enfin poser le problème, et sa solution, comme l'évoque le chapitre 3 en introduisant le répartiteur.
- L'impact des problèmes d'ordonnancement a également été observé en pratique et identifié en HIL. Le phénomène observé lors du non-respect de la précedence dans le séquençement des modules est notamment une non-réactivité du système (i.e. un manque de réactivité lors d'un asservissement en cap ou lorsque l'opérateur pilote directement le robot au joystick et que celui-ci ne réagit pas directement aux consignes envoyées sur les actionneurs). Ceci a mis en évidence le besoin d'une évolution importante de ContrAct : associer une critère de précedence aux schémas, et non pas qu'aux modules. Nous avons contourné le problème en recourant à des ordonnancements statiques. Cette question sera à nouveau évoquée dans le chapitre suivant.



Figure 5-2 : Expérimentations en piscine (Jack 6)

Le répartiteur a également été testé sur la version 12 moteurs, à savoir le jack et son extension d'actionnement. La polyvalence sur l'étage d'actionnement, changement de 6 à 12 moteurs, a été validée. La Figure 5-3 montre cette expérimentation en piscine. Le répartiteur employé ici réplique la solution pour le Jack 6 aux deux étages d'actionnement horizontaux. Il ne combine pas les actionnements, comme indiqué à la section 3.5.6, et qui permet une gestion fine de la redondance. Nous avons cependant pu valider la modification minimale que ce changement de configuration matérielle occasionne sur l'architecture logicielle que nous proposons. La mise en œuvre des asservissements en roulis et tangage, et ceux relatifs au formalisme des quaternions nécessitent des ajustements mécaniques encore en cours.

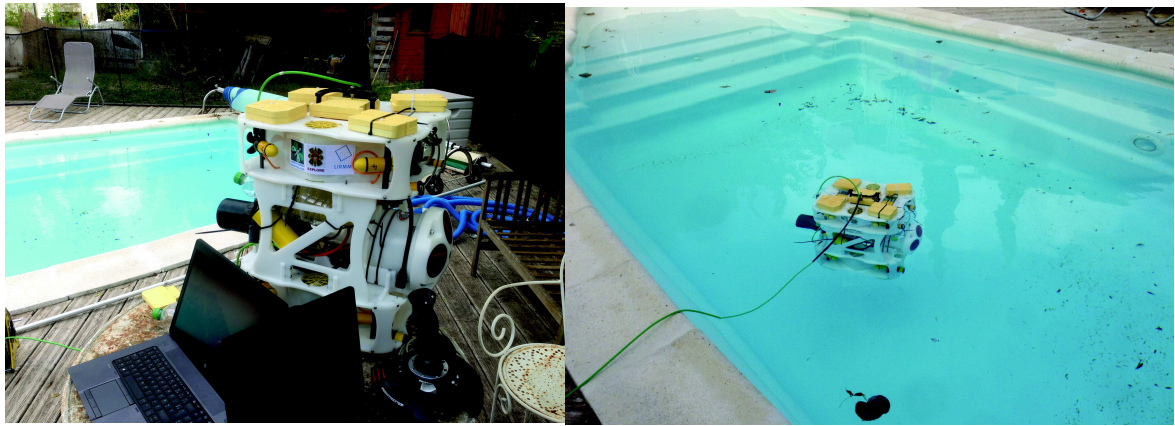


Figure 5-3 : Expérimentation du Jack 12 en piscine

5.2 Les Expérimentations terrains

5.2.1 Immersion du Jack 12 en environnement réel.

Lors des essais en piscine, nous avons rencontré des problèmes d'intégration du DVL SeaPilot de Rowe, lors de l'étude de l'extension d'actionnement. Le capteur générait de fréquentes données invalides (Figure 5-4), simplement dues à la faible profondeur de la piscine d'expérimentation. Nous avons donc déployé cette version 12 moteurs avec le DVL dans un environnement naturel (Figure 5-5) pour pouvoir valider les asservissements en vitesses, et poursuivre l'étude sur l'extension d'actionnement.

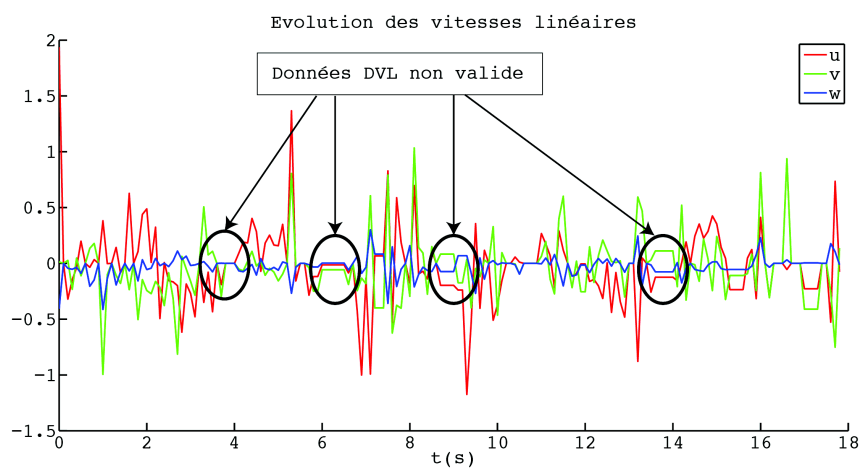


Figure 5-4 : Données DVL en piscine

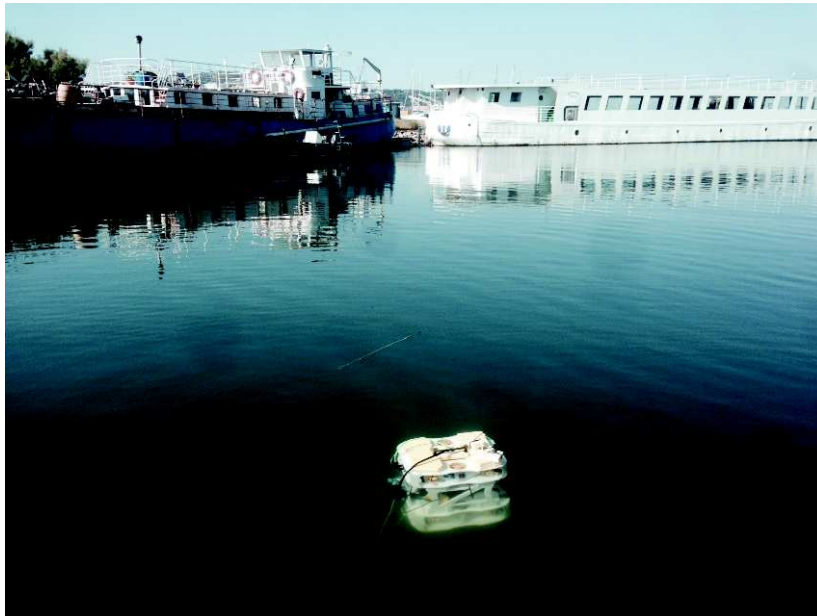


Figure 5-5 : Jack en version 12 en environnement naturel

Nous pouvons constater, sur la Figure 5-6, l'absence de données erronées, dans cet environnement suffisamment profond. Dans cette expérimentation, les 3 angles sont asservis ($\phi^d = 180^\circ$, $\theta^d = 0^\circ$ et $\psi^d = 0^\circ$) et les vitesses u et v sont asservies à 0 m/s dans un mode de « station keeping ». Dans cette manipulation, nous utilisons le compas du DVL ; or ce dernier étant mal axé cela induit un léger décalage dans entre ψ et ψ^d (Figure 5-6, pour une consigne $\psi^d = 0^\circ$, ψ est stabilisé à 350°). Il est à noter que les coefficients de corrections proposés au chapitre 3 n'ont pas été utilisés. La dérive de 0.5m en X et Y est occasionnée par l'intégration des données de vitesses issues du DVL et les imperfections de son montage.

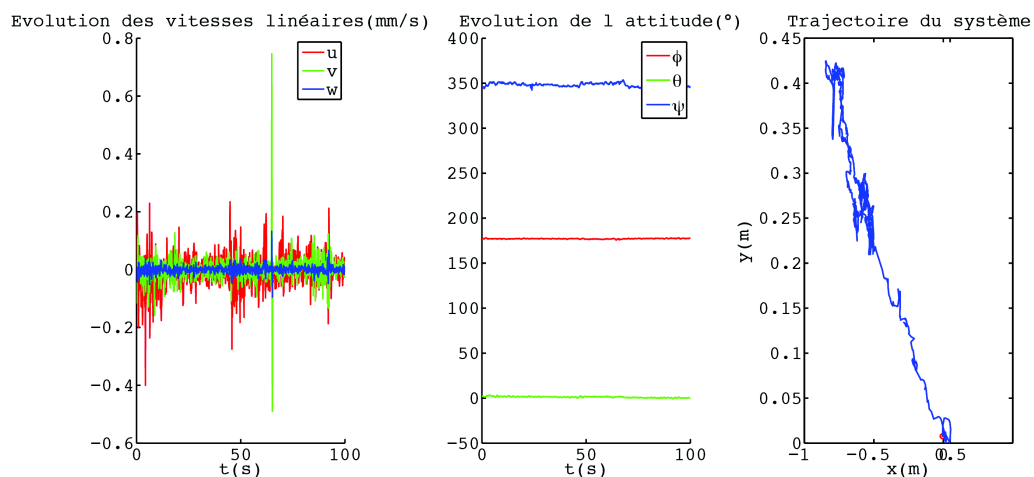


Figure 5-6 : Asservissement en vitesse du vecteur

5.2.2 Inspection de structures immergées

L'inspection de structures immergées requiert différents types de capteurs, dont une caméra acoustique donnant une image du plan horizontal, dans le sens de progression du vecteur. Dans le cadre de la validation de l'exploitation de ce capteur et de l'étude de la manœuvrabilité du Jack pour venir inspecter des zones d'intérêt, plusieurs expérimentations ont été conduites. Etudier la manœuvrabilité est un point important puisque l'emport de ces capteurs encombrants nous amène à adapter la « morphologie » du vecteur ; par exemple, la caméra acoustique est embarquée en ajoutant un skid luge (Figure 5-7), non motorisé ici contrairement à la version 12 moteurs requise pour emporter plusieurs capteurs encombrants (Figure 5-3).



Figure 5-7 : Jack équipé de la caméra acoustique installé sur le skid luge

Les premiers essais ont porté sur l'inspection de structures portuaires, sur des points d'intérêt comme une digue de port et des pontons (Figure 5-8).

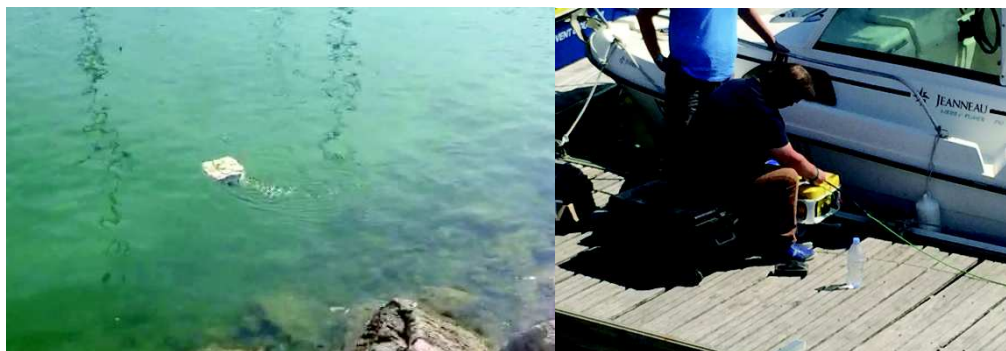


Figure 5-8 : Expérimentation d'inspection de structure portuaire

Lors de cette expérimentation le robot était équipé de la caméra Blueview²⁷ de Teledyne montée sur le skid luge (Figure 5-7). Cette caméra avec une ouverture de 90° fonctionnant à 900kHz permet d'obtenir une vue de l'environnement se trouvant devant le robot (sur une portée limitée à 100 mètres). La Figure 5-9 donne des images de la caméra acoustique dans

²⁷ <http://www.blueview.com>

lesquelles nous pouvons observer les différents rochers qui composent la digue du port. L'extraction de la distance nous permettra de faire du positionnement et/ou du suivi de paroi.

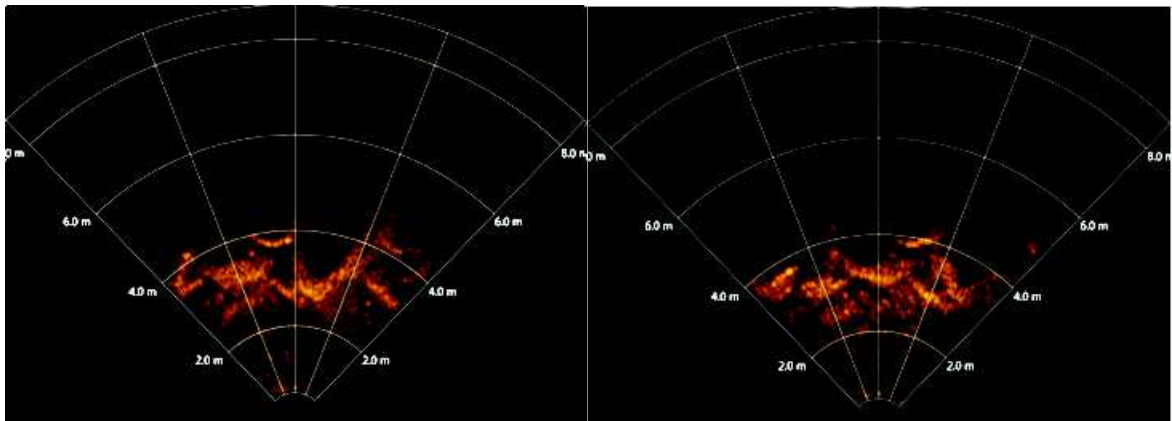


Figure 5-9 : Images obtenues avec la caméra acoustique Blueview lors de l'expérimentation.

Lors de cette expérimentation nous avons également validé le changement de mode (Figure 5-10), les différents modules utilisés dans la Figure 5-10-a sont décrits dans l'annexe I. Dans cette Figure 5-10-a, nous passons du mode par défaut (en bleu) au mode asservissement en cap (en rose), i.e. seul le cap est asservi automatiquement, l'opérateur télé-opérant les vitesses linéaires u et v . Nous pouvons constater que le robot vient se positionner sur sa consigne lors de l'activation de l'asservissement (Figure 5-10-c). Nous voyons également sur la Figure 5-10-b que la vitesse de rotation (r) est proche de zéro lorsque le vecteur est sur sa consigne de cap. Il est à noter que les perturbations extérieures présentes sur le terrain réel augmentent les effets indésirables de l'ombilical.

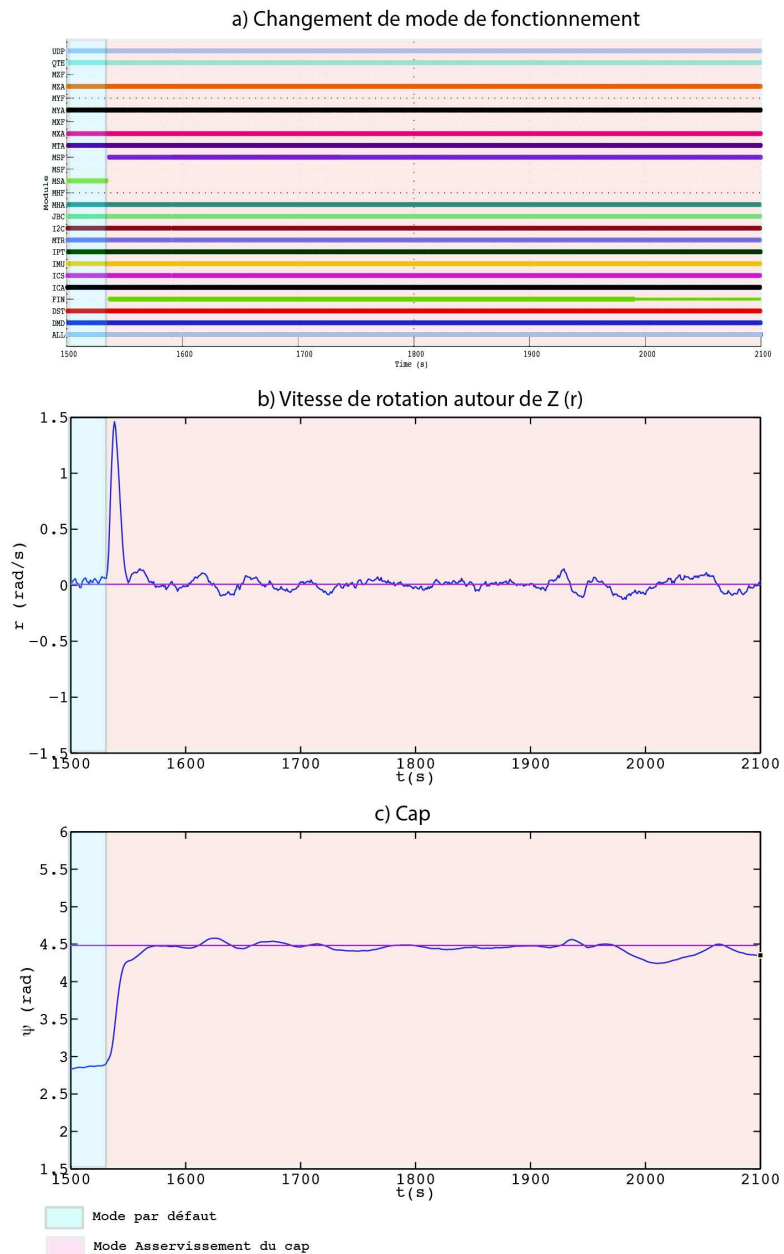


Figure 5-10 : Changement de mode de fonctionnement

5.2.3 Inspection de voie navigable

Nous avons également réalisé une expérimentation visant à caractériser l'érosion des berges d'une voie navigable, en l'occurrence du Canal du Midi. Cette érosion, appelée batillage, est induite par la motorisation des bateaux naviguant sur la canal augmentant l'effet des vagues d'étrave et de sillage, bien plus importante maintenant que lorsque le halage était employé.

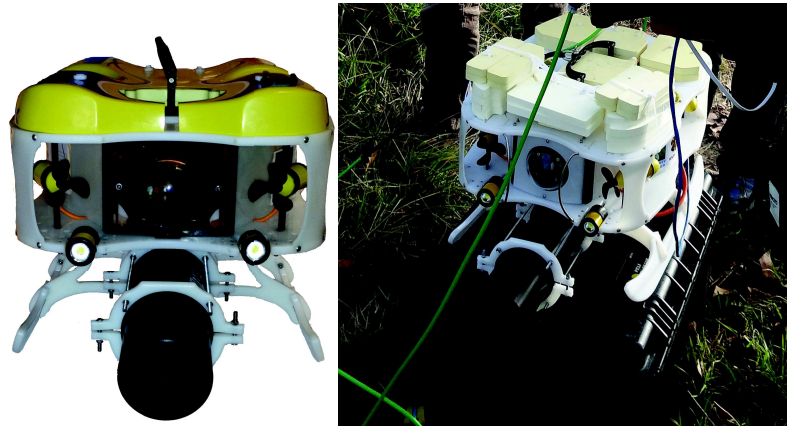


Figure 5-11 : Jack équipé du sonar profilométrique installé sur le skid luge

Pour visualiser ce phénomène, nous avons équipé le robot d'un sonar profilométrique embarqué via le skid non motorisé (Figure 5-11). Ce sonar de la marque Tritech est un sonar à balayage mécanique sur 360°, il permet de faire une insonification à différentes portées par un ajustement de la fenêtre d'écoute de la réflexion acoustique. Si cette portée est surdimensionnée elle provoquera un effet d'échos sur l'image acoustique, de par le phénomène de réflexions multiples, aussi appelé multi-chemins. Ce système permet aussi de paramétrer la fréquence d'utilisation, 600 et 1100 kHz.

Portée / Résolution	LOW	MEDIUM	HIGH	ULTRA
2m	3.77s	3.85s	4.02s	5.25s
5m	3.78s	3.87s	4.04s	6.4s
10m	3.76s	4.23s	5.9s	10.16s
15m	4.3s	5.59s	7.9s	13.4s
20m	5.65s	7.31s	10.2s	17.6s

Tableau 5-1 : Mesure du temps d'acquisition sur 360° du sonar profilométrique

Lors de cette expérimentation (Figure 5-13), le robot était asservi en cap ($\psi^d = 290^\circ$), afin de ne laisser uniquement que l'avance à la gestion de l'opérateur. Pour que le profil du canal soit d'une qualité correcte, nous avons configuré la vitesse de l'engin pour qu'elle soit constante à 0.5m/s. La Figure 5-12 retrace le cap de l'engin lors du scan du canal, nous pouvons constater un léger offset de 1° par rapport au cap désiré (ψ^d), que nous attribuons à nouveau à l'effet du câble. Il faut noter que le sonar profilométrique est du type *mechanical pencil beam sonar*. Il entraîne mécaniquement la tête acoustique (émetteur et récepteur). Ceci engendre des lenteurs en termes de réponse du capteur. Le Tableau 5-1 précise les différentes périodes d'acquisition d'un scan complet (360°) pour les différentes portées. En effet la fenêtre d'écoute et la vitesse de rotation de la tête acoustique sont couplées. Cependant, il faut noter la lenteur de ce capteur. Le centrage, tel que nous l'envisageons dans le karst, requiert une vitesse d'acquisition bien plus rapide, de l'ordre de la vitesse de la boucle de commande, 50ms ou 100ms dans notre cas, ce qui n'est pas atteignable par ce capteur, qui est pourtant l'un des plus rapide du marché. Il nous faut donc pouvoir effectuer un calcul prédictif entre deux acquisitions. Ceci requiert une connaissance des vitesses de l'engin, et donc le Loch-Doppler, et d'un modèle de l'évolution des mesures, nous reviendrons sur ce point dans le chapitre suivant.

Au moment de cette expérimentation, nous ne disposons pas encore du Loch-Doppler. Le centrage n'a donc pas pu être mis en œuvre. Nous avons cependant validé la mise en œuvre du sonar profilométrique, les algorithmes de reconstruction en ligne du modèle acquis.

La Figure 5-12 montre la tenue du cap. On constate un léger offset du à la présence du câble ombilical, tenu par un opérateur qui suivait le robot sur la berge, induisant un effet de couple positif sur le point de fixation du câble au robot.

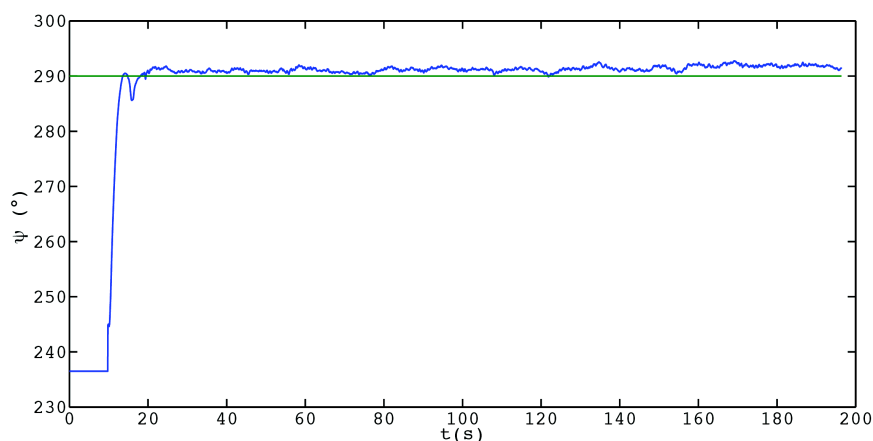


Figure 5-12 : Asservissement en cap

Le modèle 3D, reconstruit en ligne est présenté à la Figure 5-14. Nous avons réduit l'excursion du capteur profilométrique à un quart de cercle, réduisant ainsi la période d'acquisition à des durées acceptables.

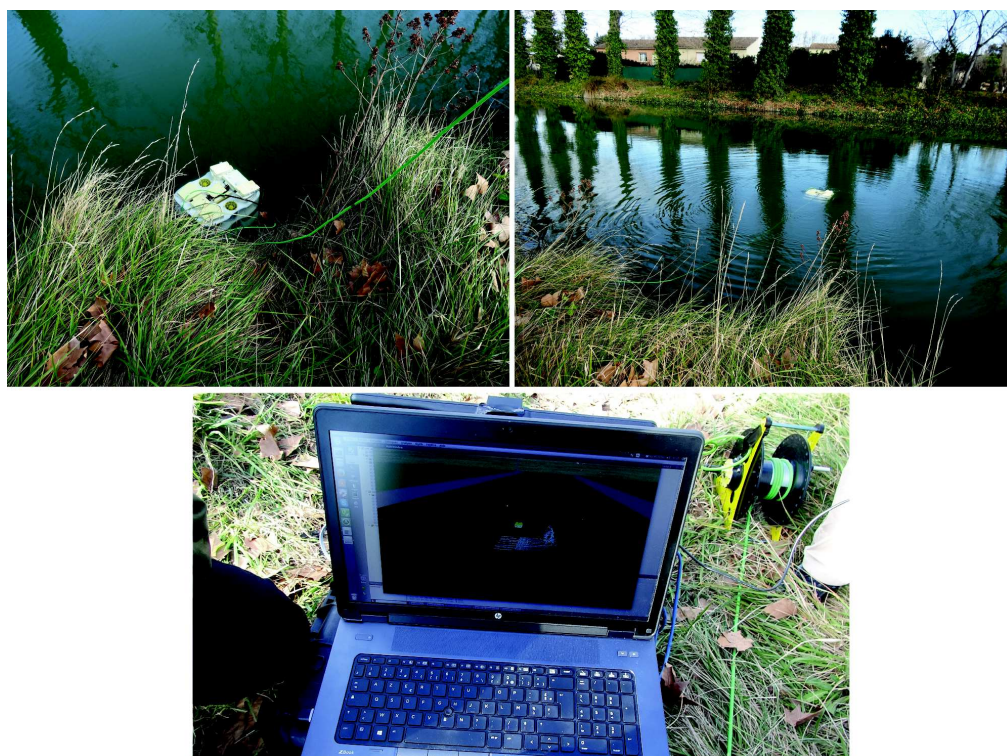


Figure 5-13 : Expérimentation au Canal du Midi

Une première analyse des données acquises montrent la pertinence de ce type d'application. Une analyse de l'écart type sur une moyenne glissante permet de détecter les artéfacts de haute fréquence présents dans le modèle brut. Ainsi on peut estimer un état de l'érosion des berges, comme le montre la figure 14.

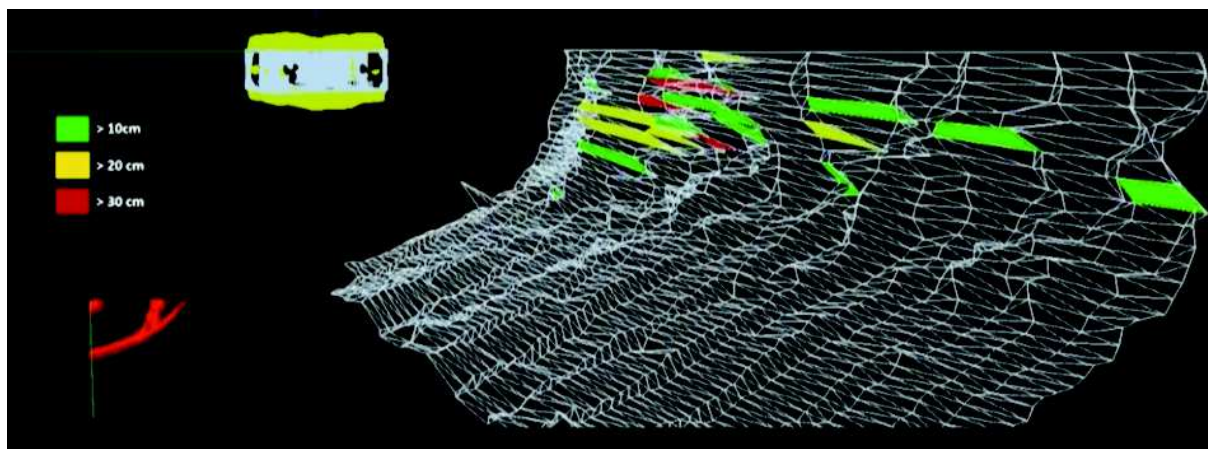


Figure 5-14 : Profil d'une berge du canal

5.2.4 Suivi d'écosystème

Cette version du robot, équipée du sonar profilométrique embarqué sur un skid non motorisé, a également été déployée sur un cimetière marin dans lequel la présence d'épaves favorise le développement et la protection de la faune (du fait de la présence de ces nombreux récifs artificiels). Des caméras Go-Pro ont été ajoutées au robot afin d'améliorer la prise d'image (Figure 5-15).



Figure 5-15 : Le robot Jack équipé du sonar profilométrique et des caméras Go-Pro

Cette expérimentation a été réalisée en collaboration avec des biologistes marins dans le cadre d'un partenariat avec le laboratoire de recherche MARBEC²⁸ (Biodiversité Marine,

²⁸ <http://www.umr-marbec.fr/ft/>

Exploitation et Conservation). Le but de cette mission était de réaliser des transects en validant les lois de commande réalisé par S. Louis dans le cadre de sa thèse. Le transect consiste à suivre d'une référence virtuelle (ligne virtuelle) permettant de visualiser l'évolution de la faune et de la flore d'un récif ou d'une structure immergée (Figure 5-16) ; deux asservissements ont été activés, l'auto-cap et l'auto-profondeur, alors que l'opérateur télé-opérait l'avance.



Figure 5-16 : Images prises lors de cette expérimentation avec les biologistes.

L'utilisation du sonar profilométrique en faible résolution, et une vitesse d'avance trop rapide à une telle résolution, n'a permis qu'une reconstruction grossière (en 3 dimensions) d'une des épaves servant de récif artificiel pour la faune et la flore locales (Figure 5-17). Pour être réellement pertinente, ces données terrain devraient être enrichies des textures acquises par la caméra du robot, si la turbidité de l'eau le permet. Ces développements sont à venir.

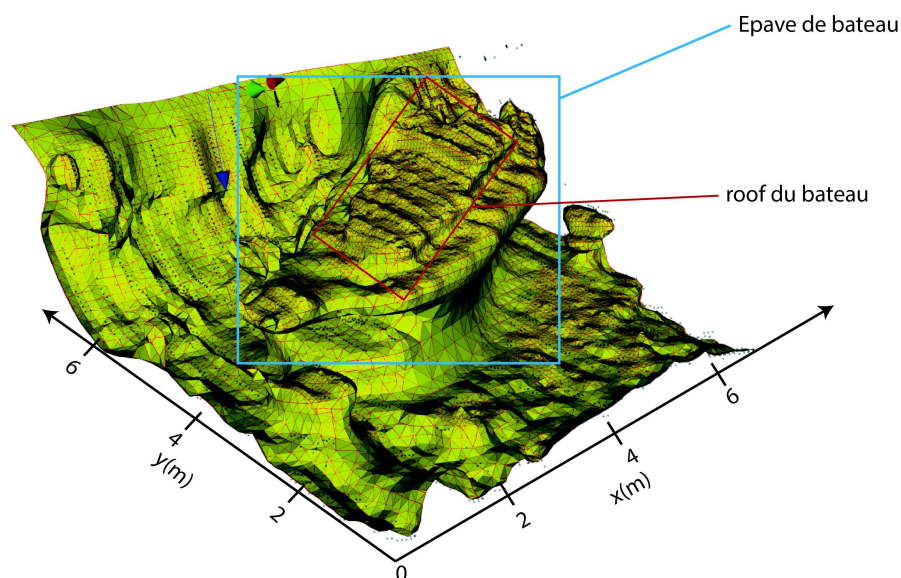


Figure 5-17 : Récif artificiel reconstruit à l'aide des données sonars

5.2.5 Archéologie

Le robot a également été utilisé en Bolivie dans le cadre du projet archéologique « Huiñaimarca » en partenariat avec l'Université Libre de Bruxelles (ULB) afin de réaliser des prospections sur des sites contenant des vestiges de la civilisation Inca (Figure 5-18).

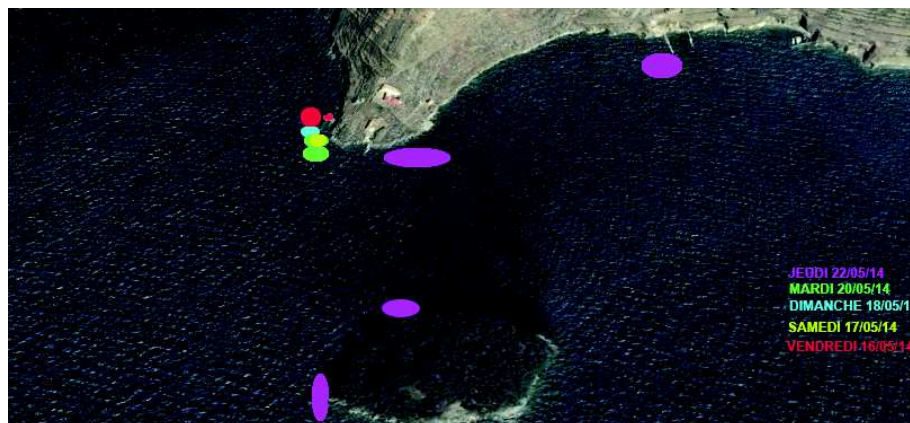


Figure 5-18 : Zone de prospection dans le lac Titicaca (Bolivie)

Le vecteur a été télé-opéré en mode cap et profondeur asservis. Le robot a été utile dans le cadre de cette expérimentation dans la mesure où il a permis de limiter le nombre et la durée des plongées des archéologues. En effet, les plongées sont contraintes sur ce site qui se situe à une altitude élevée, celle du lac Titicaca à 3812m. En plus de faire de la prospection, le robot était équipé d'une paire stéréoscopique pour relever des données visuelles pouvant être exploitées pour faire de la reconstruction 3D du fond (les objets étant souvent disposés sur les sédiments). La reconstruction 3D des fonds filmés par notre système d'acquisition n'a pu être effectuée. En effet pour diminuer l'absorption des ondes électromagnétiques de la lumière laser par l'eau, nous avons choisi des lasers de couleur violette. Malheureusement, les filtres internes des caméras Go-Pro et l'égalisation chromatique automatique effectuée par les caméras n'ont pas permis d'obtenir des images dans lesquelles les 4 raies laser présentaient un contraste suffisant pour permettre la reconstruction.

Cette mission a été aussi l'occasion de démontrer nos capacités opérationnelles pour déployer notre système sur des sites lointains.

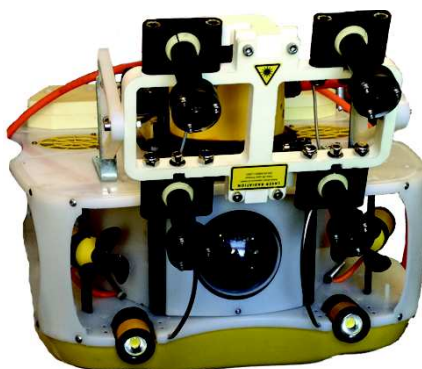


Figure 5-19 : Le robot Jack équipé d'une paire de caméras supplémentaires



Figure 5-20 : Le jack dans le cadre des recherches archéologiques en Bolivie

5.3 Expérimentations à venir

Les prochaines expérimentations que nous allons réaliser vont permettre de poursuivre la validation de nos propositions.

5.3.1 Mayotte, novembre/décembre 2015

En cette fin d'année le robot part réaliser des missions de terrain en partenariat avec le laboratoire de recherche MARBEC dans le lagon de Mayotte, en collaboration avec nos collègues biologistes marins. La mission sera pilotée par Silvain Louis et va permettre de valider les modes de fonctionnement du robot, en particulier le transect. Le formalisme quaternion est utilisé. Un autre objectif de cette mission est de tester les effets de la présence du robot sur la qualité du comptage de poissons. Les protocoles expérimentaux ont été établis avec nos partenaires. Des comparaisons de résultats de comptage avec les moyens classiques (plongeurs) et avec le robot seront effectuées afin de valider notre approche.

D'autres missions sont programmées pour l'année 2016. Elles sont décrites au chapitre suivant.

5.4 Points clefs

- Ces nombreuses expérimentations terrains ont permis de développer plusieurs versions du vecteur ainsi que de valider les différents concepts évoqués dans cette thèse.
- Grâce à ces expérimentations, nous pouvons constater le large spectre des applications réalisables par le robot.
- La partie expérimentation n'est pas la partie la plus simple à mettre en œuvre, car la charge logistique qu'ils occasionnent nécessite un travail préparatoire conséquent.

- Cette partie est l'achèvement de toute notre démarche dans la recherche de la polyvalence.

Conclusion et Perspectives

Nos travaux portent sur la polyvalence d'un vecteur sous-marin de petites dimensions, dédié à l'exploration d'environnements sub-aquatiques faible fond ou confinés. Cette polyvalence a été étudiée sur les différents aspects du vecteur, à savoir sa structuration mécanique, ses architectures embarquées matérielle et logicielle ainsi que sa commande. Ses travaux ont été appliqués au robot sous-marin « Jack » développé par le partenaire industriel Ciscrea.

Le besoin de polyvalence nous a conduit à complètement repenser l'architecture matérielle (électronique) du robot ; elle est basée sur un réseau Ethernet embarqué facilitant la modification de la charge utile, à savoir l'emport de différents capteurs acoustiques selon l'application à réaliser. Cette architecture permet également l'évolution du vecteur par ajout de caissons étanches tels qu'un caisson d'actionnement de 6 moteurs (skid actionné, passant le nombre de moteurs de 6 à 12), ou encore un caisson dédié au contrôle évolué (puissance de calcul supplémentaire).

Une nouvelle architecture embarquée a également été proposée sur le plan logiciel. Cette nouvelle architecture de contrôle a été conçue avec comme préoccupation une modularité et une structuration adéquates à la polyvalence recherchée. La modularité est en effet indispensable pour faire évoluer aisément le vecteur robotique afin de pouvoir répondre aux besoins métiers, comme par exemple intégrer de nouveaux capteurs (caméra acoustique, sonar profilométrique,...), ajouter des propulseurs (dans la version skid actionné) et modifier le contrôle, et ainsi s'adapter à l'application à réaliser. Cette architecture logicielle de contrôle, basée sur une approche à services et une gestion dynamique de ressources abstraites et de modules temps-réel, a été prototypée en s'appuyant sur le middleware temps-réel ContrAct du LIRMM.

Viser la polyvalence n'impacte pas seulement les architectures matérielle et logicielle, mais également le contrôle lui-même. Au delà des différentes lois de commande, l'enjeu se situe dans la gestion de l'actionnement. Par exemple, comment éviter de réécrire toutes les lois de commande quand le vecteur passe d'une version 6 moteurs à une version 12 moteurs ? Cela a nécessité une étude approfondie de l'actionnement, et de sa redondance. La proposition réside dans l'abstraction de l'actionnement par l'intermédiaire d'un module appelé répartiteur. Ce dernier permet une gestion fine de la redondance d'actionnement à des fins de robustesse et de pilotage de la réactivité de l'actionnement. Plusieurs perspectives s'ouvrent quant à la finalisation de cette étude :

- Robustesse à la disparité des moteurs : établissement des coefficients de correction

Pour le Jack 6, l'exploration de l'espace de redondance est aisée, de par le fait qu'il est de dimension 1. Pour l'actionnement horizontal du Jack 12, la redondance est de degré 3. Une exploration finement discrétisée de cet espace requiert un trop grand nombre d'essais pour être pratiquement réalisable. L'approche suivie pour le Jack 6 ne permet pas d'exciter les disparités haut/bas (HB) et pair/impair (PI). Elle consiste à asservir la position du système autour de l'origine, ce qui peut se faire indépendamment des disparités HB et PI. Il faut pouvoir exciter ces disparités durant les asservissements, trouver des trajectoires d'excitation réalisables.

- Considération des incertitudes de positionnement des moteurs

Nous n'avons pas traité explicitement de la question de l'erreur de positionnement des moteurs. Ceci revient à considérer un répartiteur construit de la façon décrite à l'équation (6.1).

$$\mathbb{R} = [\widehat{\mathbb{R}}_G \quad \mathbf{M}_M] \quad (6.1)$$

où $\widehat{\mathbb{R}}_G = \widehat{\mathbb{C}}^+$, $\widehat{\mathbb{C}}^+$ désignant la pseudo-inverse de $\widehat{\mathbb{C}}$, une estimation de \mathbb{C} . Il faut noter que cette erreur est paramétrique (constante), au même titre que les caractéristiques moteur. Nous avons montré que, dans le cas linéaire, il existe une procédure d'étalonnage pratique qui conduit à l'identification d'une matrice de coefficients pour corriger les effets de la disparité des caractéristiques actionneurs. Il est intéressant de voir si la procédure en question est en mesure de corriger certains effets de l'erreur de positionnement des moteurs. Nous réalisons une simulation de cette question en bruitant les positions réelles des moteurs, comme indiqué dans le Tableau 3-4 pour le Jack 6. Nous procédons à la procédure d'étalonnage, pour obtenir les coefficients de correction de la Figure 6-1-a. Nous comparons ensuite la trajectoire du système, suite à une consigne \mathbf{F}_B^d , avec (Figure 6-1-b) et sans application de la matrice de correction \mathbb{Q} (Figure 6-1-c).

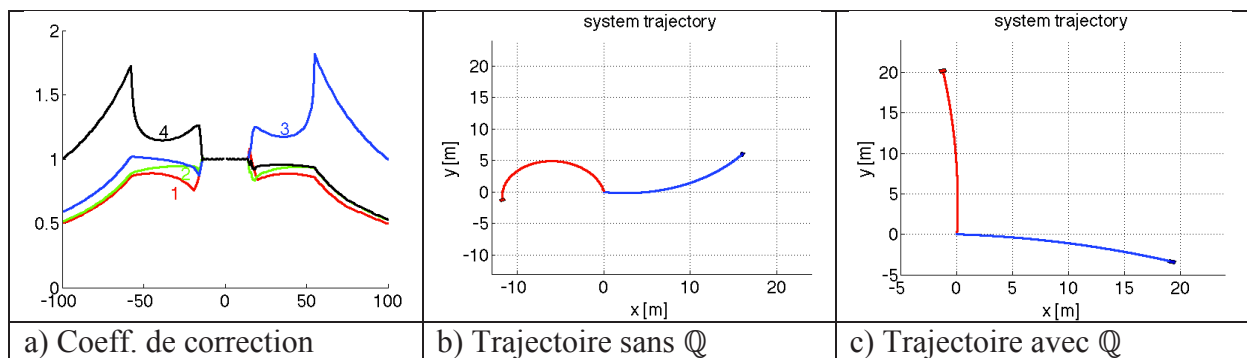


Figure 6-1 : Evaluation de la prise en compte des incertitudes de positionnement des moteurs.

Les améliorations sont nettes, mais pas parfaites. L'approche théorique dans le cadre de notre approche reste à établir.

➤ Considération simultanée des différentes méthodes

Nous avons vu que la redondance peut être exploitée avec des objectifs différents. Nous n'avons pour l'instant traité des objectifs de robustesse et de tolérance aux fautes que de manière indépendante. Il serait intéressant de définir un cadre commun dans lequel les précédentes méthodes puissent être simultanément considérées.

➤ Identification du modèle dynamique

L'étude du modèle dynamique dont l'expression générale est rappelée ci-dessous, permet d'envisager une procédure d'identification qui s'inspire de celle effectuée pour la

considération des disparités des caractéristiques des moteurs. Afin de dissocier les différentes causes, nous considérons les caractéristiques et les positions des moteurs comme connues.

$$\begin{aligned}
F_u &= X_{\dot{u}} \cdot \dot{u} + X_{u,|u|} \cdot u \cdot |u| + X_{w,q} \cdot w \cdot q + X_{v,r} \cdot v \cdot r + X_G(\boldsymbol{\eta}_U) \\
F_v &= Y_{\dot{v}} \cdot \dot{v} + Y_{v,|v|} \cdot v \cdot |v| + Y_{u,r} \cdot u \cdot r + Y_{w,p} \cdot w \cdot p + Y_G(\boldsymbol{\eta}_U) \\
F_w &= Z_{\dot{w}} \cdot \dot{w} + X_{w,|w|} \cdot w \cdot |w| + Z_{u,q} \cdot u \cdot q + Z_{v,p} \cdot v \cdot p + Z_G(\boldsymbol{\eta}_U) \\
\Gamma_p &= K_{\dot{p}} \cdot \dot{p} + K_{p,|p|} \cdot p \cdot |p| + K_{q,r} \cdot q \cdot r + K_v \cdot v + K_w \cdot w + K_G(\boldsymbol{\eta}_U) \\
\Gamma_q &= M_{\dot{q}} \cdot \dot{q} + M_{q,|q|} \cdot q \cdot |q| + M_{p,r} \cdot p \cdot r + M_u \cdot u + M_w \cdot w + M_G(\boldsymbol{\eta}_U) \\
\Gamma_r &= N_{\dot{r}} \cdot \dot{r} + N_{r,|r|} \cdot r \cdot |r| + N_{p,q} \cdot p \cdot q + N_u \cdot u + N_v \cdot v + N_G(\boldsymbol{\eta}_U)
\end{aligned}$$

Si comme fait précédemment, un asservissement PID des 6 vitesses (rotationnelles et linéaires) est effectué autour de 0, les moteurs atteindront un régime statique qui comprendra les effets des paramètres de flottabilité $[X_G(0), Y_G(0), Z_G(0), K_G(0), M_G(0), N_G(0)]$. Ce qui permet d'identifier la position du métacentre (barycentre entre le centre de flottabilité et le centre de gravité). Une fois ces paramètres identifiés, une série de tests avec des consignes d'asservissement ne sollicitant qu'un axe à la fois (par exemple, un asservissement qui permet le seul déplacement selon l'axe u , avec pour consigne une vitesse désirée u_d) permettra d'identifier les paramètres de couplages dus à la disparité des symétries de formes de l'engin, et ceux de trainée. (M_u, N_u et $X_{u,|u|}$, pour l'exemple évoqué précédemment), par l'analyse des régimes statiques atteints par les moteurs. De même un asservissement croisé des vitesses linéaires et rotationnelles permettra d'identifier les termes de couplages. La masse ajoutée est plus difficile à estimer pratiquement. Notons cependant que les applications que nous envisageons seront souvent réalisées avec des vitesses constantes, ne sollicitant donc pas les termes inertiels. Reste à établir les protocoles expérimentaux, ce qui peut devenir fastidieux suivant les capteurs dont on dispose.

➤ Adaptation et robustesse

La robustesse dont nous avons traité dans ce manuscrit est une méthode d'identification préalable qui 'rééquilibre' la boucle ouverte. Cependant, l'identification n'est possible que si nous disposons de capteurs en mesure de réaliser les asservissements nécessaires dans des configurations particulières. L'avantage est qu'ensuite, une commande en boucle ouverte 'équilibrée' (découplée) est possible sans ces capteurs. Il serait cependant intéressant d'adapter la procédure d'identification préalable en un schéma adaptatif qui serait exécuté en ligne.

Les perspectives portent également sur l'architecture logicielle de contrôle. L'approche que nous avons proposée facilite l'évolution de l'architecture embarquée et la mise en place de différents modes de fonctionnement (services) par composition de briques logicielles (modules), en assurant une composition cohérente par la réification des ressources et une exploitation « transparente » de l'actionnement grâce à son abstraction. Néanmoins, nous avons mis en exergue dans cette étude l'impact du middleware temps-réel dans le déterminisme et la réactivité de l'application. Ainsi, pour le middleware retenu (ContrAct), nous avons identifié un ensemble d'évolutions nécessaires relatives à la performance de commutation dynamique de compositions (i.e., services), telles que l'ordonnancement non plus seulement au sein d'une composition (schéma) mais entre compositions (inter-schémas).

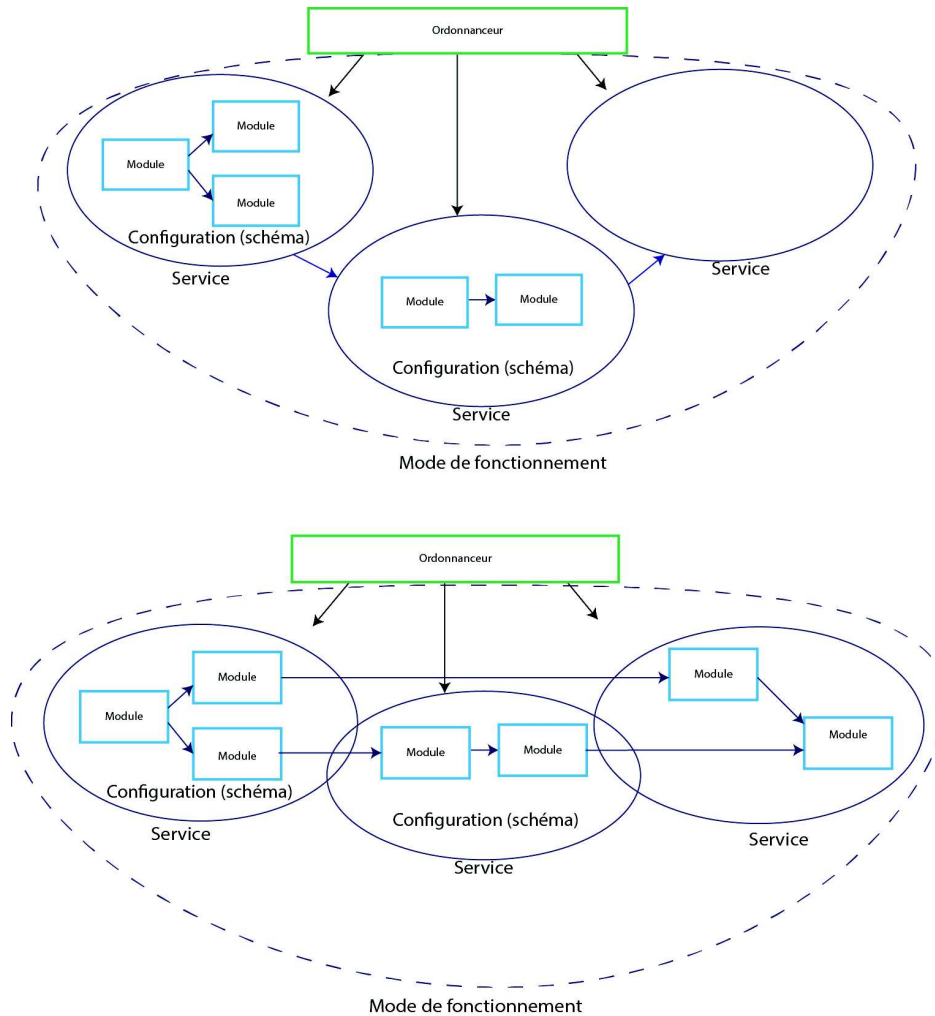


Figure 6-2 : Ordonnancement inter-schémas

La commutation de compositions ne se résume pas à activer ou à désactiver des modules au sein de schémas, elle doit nécessairement gérer le contrôle du robot durant cette phase. Il s'agit là d'une perspective importante que nous pourrions aborder en exploitant les travaux de thèse de A. Lasbouygues[40] dont le formalisme basé atomes en permet une spécification « implémentable » (i.e., du modèle à sa projection sur le middleware temps-réel).

Au bilan, nous avons conceptualisé, développé (mécanique, électronique, automatique, informatique) et expérimenté la polyvalence. Tous ces travaux ont en effet nécessité de nombreux développements, et de nombreuses expérimentations ont été réalisées, en piscine et sur sites (canal et port). Nous avons constaté par le biais de ces expérimentations, au fil des évolutions du vecteur et de l'intégration de capteurs de charge utile, que le spectre d'applications offert par ce vecteur est grand. Une application 'phare' pour nos recherches, tant de par les défis scientifiques et technologiques que des enjeux socio-économiques, est l'exploration de réseaux karstiques. Une telle exploration, seule issue pour aller au delà des limites humaines (des plongeurs spéléologues) dans la découverte et la compréhension des ressources d'eau douce souterraines, requiert notamment le développement de nouveaux types de capteurs et de stratégies et lois de commande avancées. Nos travaux ont conduit à un vecteur qui, de par sa polyvalence et son évolutivité, permettra à ces recherches d'être conduites en faisant « disparaître » le vecteur au profit d'un ensemble de capteurs à mobilité

contrôlée : tel est le défi de l'exploration des environnements subaquatiques confinés, amener des capteurs là où l'homme veut acquérir de la connaissance.

Nous avons mis en évidence les contraintes qu'apportent l'utilisation des capteurs du commerce, notamment le sonar profilométrique. En effet ce dernier présente des périodes d'acquisition qui ne permettent pas l'utilisation directe des mesures dans la boucle de commande. Une phase prédictive est nécessaire afin de compléter les mesures dans les phases 'aveugles' du sonar. Ainsi cette prédiction se base sur un modèle d'environnement et l'exploitation des mesures des vitesses linéaires fournies par le Loch-Doppler. Une connaissance *a priori* est ici précieuse. En première approche, les modèles 'au premier ordre' seraient un cylindre orienté pour le karst, un demi-cylindre pour le cas du canal. Il faut noter que cette approche induit une hypothèse forte sur l'environnement. Si la prise de risque est faible pour le cas du canal, elle ne l'est plus pour le karst. En effet, on peut considérer que la période 'aveugle' durant laquelle le contrôle du système repose sur le modèle d'environnement qu'on se donne avec une bonne estimation de ses vitesses propres. On peut ainsi redéfinir l'hypothèse sous la forme d'une 'granularité spatiale' attribuée à l'environnement. C'est un pari osé pour le cas du karst, où stalactites, stalagmites, diaclases abruptes et formes saillantes sont probables. Ainsi le cas du karst requiert aussi d'avoir une image pertinente, et fréquente, de 'l'espace navigable' qui se présente dans la direction du mouvement. Le système doit donc aussi embarquer une caméra acoustique et un algorithme qui permettra de guider automatiquement le système vers les zones de navigabilités faisables – l'opérateur en sélectionnant une – ou optimale dans la situation autonome.

La validation expérimentale de tels travaux nécessite un programme détaillé, qui permet d'incrémenter la difficulté tout en maîtrisant la prise de risque. Ainsi, les missions de terrain à venir, outre celle de Mayotte évoquée précédemment, sont brièvement présentées ci-dessous.

➤ Cartographie de Gourneyras(mai 2016)

Le Gouffre de Gourneyras est un haut lieu de la plongée souterraine en Europe. Il dispose d'une grande salle d'entrée (diamètre de 36m), dans laquelle débouche un vaste réseau karstique que les plongeurs explorent régulièrement²⁹. L'exploration de la salle d'entrée par notre robot permettra de valider l'utilisation du sonar profilométrique à 360° et la prédiction issues des données du Loch-Doppler via un modèle d'environnement sphérique.

La particularité de cette mission est qu'elle sera appuyée par une équipe de spéléo-plongeurs de l'association Plongée Sout³⁰. La gestion de la charge logistique que cela occasionne sera un élément déterminant de la réussite de la mission.

➤ Cartographie de la chambre de pompage de la source du Lez (Juillet 2016)

La source du Lez est un élément clef de l'alimentation en eau de l'agglomération montpelliéraine (Montpellier Métropole). Une exploration humaine devient problématique, à cause des profondeurs rencontrées. Ainsi nous envisageons d'effectuer la cartographie de la chambre de pompage à l'aide de notre système. L'intérêt est d'effectuer un relevé de la géomorphologie de ce conduit karstique sur une zone déjà connue, nous permettant de comparer

²⁹ Voir la vidéo de la plongée de Erster Testtauchgang, juillet 2015 : <https://vimeo.com/134537949>

³⁰ <http://www.plongeesout.com>

nos résultats avec un modèle terrain disponible. De plus, lors de cette mission, le robot ne pourra être accompagné par des plongeurs, si ce n'est au début de la mission. A nouveau, la question de l'accès au site et du système de mise à l'eau alourdit considérablement les aspects logistiques.

➤ Exploration des sources du Lez (2016-2017)

La mission précédente préfigure l'exploration des zones inconnues de l'aquifère du Lez. En particulier les dernières plongées humaines ont permis de confirmer la présence d'une vaste salle dont l'entrée se situe aux environs des 80m de profondeur. Il est essentiel pour les gestionnaires du service des eaux de Montpellier Métropole d'avoir une caractérisation pertinente des dimensions de cette salle, laissant présager une capacité de pompage plus importante. La mission précédente aura permis de repérer la configuration de l'entrée de cette salle. Ainsi pour la première fois, notre robot sous-marin réalisera une véritable exploration d'un réseau karstique inconnu. La mission se cantonnera cependant à la cartographie de cette salle.

➤ Exploration de la résurgence de Font Estramar, à huit ans

Un autre célèbre lieu de plongée souterraine est localisé dans les Pyrénées Orientales, à Font Estramar. De nombreuses plongées ont été effectuées dans ce réseau et des records du monde y ont été établis (-248 en août 2013 et -262 en juillet 2015 par Xavier Méniscus³¹). Les membres de l'Association de Recherche de Font Estramar (ARFE³²) se sont déclarés disponibles pour appuyer nos travaux. Nul doute que leur aide sera précieuse, tant pour leur expertise du terrain que pour l'appui logistique qu'ils apportent.

³¹ <http://france3-regions.francetvinfo.fr/languedoc-roussillon/pyrenees-orientales/le-record-du-monde-de-plongee-souterraine-battu-salses-le-chateau-par-262-metres-770081.html>

³² <http://www.plongeesout.com/sites/roussillon-pyrenees/pyrenees%20orientales/font%20estramar%20res.htm>

Bibliographie

- [1] V. Rigaud, “Innovation and operation with robotized systems,” *J. F. Robot.*, vol. 24, no. 6, pp. 44–51, 2009, DOI:10.3182/20090916-3-BR-3001.0076.
- [2] J. Michel, M. Klages, F. J. A. S. Barriga, Y. Fouquet, S. Myriam, P.-M. Sarradin, P. Siméoni, and J.-F. Drogou, “Victor 6000: Design, Utilization and First Improvements,” in *International Offshore and Polar Engineering Conference*, 2003, vol. 5, pp. 7–14, Honolulu, USA.
- [3] P.-M. Sarradin, K. O. Leroy, H. Ondreas, M. Sibuet, M. Klages, Y. Fouquet, B. Savoye, J.-F. Drogou, and J.-L. Michel, “Evaluation of the first year of scientific use of the French ROV Victor 6000,” in *Proceedings of the 2002 International Symposium on Underwater Technology (Cat. No.02EX556)*, 2002, no. april, pp. 11–16, Tokyo, Japan, DOI:10.1109/UT.2002.1002370.
- [4] M. Caccia, M. Bibuli, R. Bono, G. Bruzzone, G. Bruzzone, and E. Spirandelli, “Unmanned marine vehicles at CNR-ISSIA,” in *The International Federation of Automatic Control*, 2008, vol. 17, no. 1, pp. 3070–3075, Seoul, Koera, DOI:10.3182/20080706-5-KR-1001.0164.
- [5] H. Nakajoh, H. Osawa, T. Miyazaki, K. Hirata, T. Sawa, and H. Utsugi, “Development of work class ROV applied for submarine resource exploration in JAMSTEC,” in *Oceans 2012*, 2012, Yeosu, Korea, DOI:10.1109/OCEANS-Yeosu.2012.6263437.
- [6] T. Murashima, H. Nakajoh, H. Yoshida, J. N. Yamauchi, and H. Sezoko, “7.000m class ROV KAIKO7000,” in *Oceans 2004*, 2004, vol. 2, pp. 812–817, Kobe, Japan, DOI:10.1109/OCEANS.2004.1405558.
- [7] A. M. Tahir and J. Iqbal, “Underwater robotic vehicles: latest development,” *Sci.Int.(Lahore)*, vol. 26, no. 3, pp. 1111–1117, 2014.
- [8] J. Rife and S. M. Rock, “A pilot-aid for ROV based tracking of gelatinous animals in the midwater,” in *MTS/IEEE Oceans 2001. An Ocean Odyssey. Conference Proceedings (IEEE Cat. No.01CH37295)*, 2001, vol. 2, pp. 1137–1144, Honolulu, USA, DOI:10.1109/OCEANS.2001.968274.
- [9] T. Salgado-Jimenez, J. L. Gonzalez-Lopez, J. C. Pedraza-Ortega, L. G. Garcia-Valdovinos, L. F. Martinez-Soto, and P. a. Resendiz-Gonzalez, “Design of ROVs for the Mexican power and oil industries,” in *2010 1st International Conference on Applied Robotics for the Power Industry (CARPI 2010)*, 2010, pp. 1–8, Montreal, Canada, DOI:10.1109/CARPI.2010.5624437.
- [10] T. Salgado-Jimenez, J. L. Gonzalez-Lopez, L. F. Martinez-Soto, E. Olguin-Lopez, P. a. Resendiz-Gonzalez, and M. Bandala-Sanchez, “Deep water ROV design for the Mexican oil industry,” in *OCEANS 2010*, 2010, pp. 1–6, Sydney, Australia, DOI:10.1109/OCEANSSYD.2010.5603516.
- [11] M. Jacobi and D. Karimanzira, “Underwater pipeline and cable inspection using autonomous underwater vehicles,” in *OCEANS 2013*, 2013, pp. 1–6, Bergen, Norway, DOI:10.1109/OCEANS-Bergen.2013.6608089.

- [12] S. Reed, J. Wood, and C. Haworth, “The detection and disposal of IED devices within harbor regions using AUVs, smart ROVs and data processing/fusion technology,” in *2010 International WaterSide Security Conference*, 2010, pp. 1–7, Marina di Carrara, Italy, DOI:10.1109/WSSC.2010.5730276.
- [13] A. W. Stoner, C. H. Ryer, S. J. Parker, P. J. Auster, and W. W. Wakefield, “Evaluating the role of fish behavior in surveys conducted with underwater vehicles,” *Can. J. Fish. Aquat. Sci.*, vol. 65, no. 6, pp. 1230–1243, Jun. 2008, DOI:10.1139/F08-032.
- [14] N. M. Bacheler, T. J. Paoli, and G. M. Schacht, “Controls on Abundance and Distribution of Yellow Perch: Predator, Water Quality, and Density-Dependent Effects,” *Trans. Am. Fish. Soc.*, vol. 140, no. 4, pp. 989–1000, 2011, DOI:10.1080/00028487.2011.603979.
- [15] S. L. Harter and A. W. David, “Examination of proposed additional closed areas on the West Florida Shelf a report to the Gulf of Mexico Fishery Management Council,” 2009.
- [16] “VICTOR 6000,” *flotte.ifremer.fr*, 2010. [Online]. Available: <http://flotte.ifremer.fr/fleet/Presentation-of-the-fleet/Underwater-systems/VICTOR-6000>.
- [17] “Coral Ecosystem Connectivity 2013,” *oceanexplorer.noaa.gov*, 2013. [Online]. Available: <http://oceanexplorer.noaa.gov/explorations/13pulleyridge/welcome.html>.
- [18] “SharkCam,” *www.whoi.edu*, 2015. [Online]. Available: <http://www.whoi.edu/osl/sharkcam>.
- [19] “TurtleCam,” *www.whoi.edu*, 2015. [Online]. Available: <http://www.whoi.edu/osl/turtlecam>.
- [20] N. a. Cruz and A. C. Matos, “The MARES AUV, a modular autonomous robot for environment sampling,” *Ocean. 2008*, 2008.
- [21] D. G. Walker, “XAUv: Modular High Maneuverability Autonomous Underwater Vehicle,” Thesis for the degree of master of science in mechanical engineering at the massachusetts institute of technology under the supervision of F. Hover and Pr. J. Leonard, Cambridge, USA, 2008.
- [22] Mi. Ma, “New tool monitors effects of tidal, wave energy on marine habitat,” *www.washington.edu*, 2015. [Online]. Available: <http://www.washington.edu/news/2015/02/05/new-tool-monitors-effects-of-tidal-wave-energy-instruments-on-marine-habitat/>.
- [23] B. Rush, J. Joslin, B. Polagye, and A. Stewart, “Development of an adaptable monitoring package for marine renewable energy projects,” in *Marine Energy Technology Symposium, METS2014*, 2014, Seattle, USA.
- [24] J. Joslin, E. Celkis, B. Polagye, and A. Stewart, “Development of an adaptable monitoring package for marine renewable energy,” in *OCEANS 2013*, 2013, San Diego, USA.
- [25] H. Gosselin, “Un robot montpelliérain pour explorer les fonds du lac Titicaca,” *La marseillaise (6 Febr. 2014)*, vol. 21004, 2014.
- [26] M. L’Hour and V. Creuze, “The 2014 underwater archaeological mission on the shipwreck of the Lune, in the framework of the Corsaire Concept Project: scanning and

- sampling artefacts with robots,” in *EUROCAST 2015 Conference - Workshop on Marine Sensors and Manipulators*, 2015, Las Palmas de Gran Canaria, Spain.
- [27] Henri Salvayre, *Le livre des eaux souterraines des Pyrénées catalanes*. Canet en Roussillon, 2010, ISBN: 2849741051.
- [28] S. Mokhtari, “Une mer d’eau douce sous les corbières,” *Midi Libre*, 2012. [Online]. Available: <http://www.midilibre.fr/2012/01/06/la-plus-grande-reserve-d-eau-douce-d-europe,440296.php>.
- [29] “Font Estramar : une plongée extrême à -248 m.,” *Spéléo Mag.*, vol. N°83, pp. 8–13, 2013.
- [30] R. Pastor, “Fontaine de Vaucluse,” *Société spéléologique de Fontaine de Vaucluse*. [Online]. Available: <http://www.plongeesout.com/sites/provence/vaucluse/vaucluse-fontaine.htm>.
- [31] “Sorgonaute,” *Comité départemental de spéléologie de seine-maritime*, 2009. [Online]. Available: <http://cd.speleo76.free.fr/phpwebgallery/picture.php?/1109/category/2>.
- [32] C. Emig, “Modexa,” 1985. [Online]. Available: <http://paleopolis.rediris.es/Phoronida/EMIG/Souvenirs/Navires/N-O.html>.
- [33] “Spélénaute,” *www.ssfv.fr*, 1989. [Online]. Available: <http://www.ssfv.fr>.
- [34] G. Caramanna, “Scientific Diving and ROV Techniques Applied to the Geomorphological and Hydrogeological Study of the World ’ s Deepest Karst Sinkhole , (Pozzo del Merro – Latium – Italy),” 1994.
- [35] L. Kolovrat, “Red Lake Finally Discovered Some Secrets,” *www.adria-navigator.com*, 2013. [Online]. Available: <http://www.adria-navigator.com/article/red-lake-finally-discovered-some-secrets/>.
- [36] “DEPTHX (DEep Phreatic THERmal eXplorer),” *www.stoneaerospace.com*. [Online]. Available: <http://www.stoneaerospace.com/products-pages/products-DEPTHX.php>.
- [37] M. Gary, N. Fairfield, W. C. Stone, D. Wettergreen, G. Kantor, and J. M. Sharp, Jr., “3D Mapping and Characterization of Sistema Zacatón from DEPTHX (DEep Phreatic THERmal eXplorer),” in *KARST ’08*, 2008, pp. 202–212, Reston, VA, USA, DOI:10.1061/41003(327)20.
- [38] R. Passama, “ContrACT : une méthodologie de conception et de développement d ’ architectures de contrôle de robots(Rapport LIRMM n°: RR-10025),” Montpellier, France, 2010.
- [39] R. Passama and D. Andreu, “ContrACT : A software environment for developing control architecture,” in *CAR ’11*, 2011, Grenoble, France.
- [40] A. Lasbouygues, “Exploration robotique de l’environnement aquatique : les modèles au coeur du contrôle,” Thèse de doctorat en Systèmes Automatiques et Microélectroniques, sous la direction de D. Andreu et de L. Lapiere, Université de montpellier, Montpellier, France, 2015.
- [41] SNAME The Society of Naval Architects and Marine, “Nomenclature for Treating the Motion of a Submerged Body Through a Fluid,” *Technical and Research Bulletin*, vol. 1–5. 1950.
- [42] T. I. Fossen, *Guidance and Control of Ocean vehicles*. UK: John Wiley & Sons Ltd.,

1994, ISBN: 9780471941132.

- [43] T. I. Fossen, *Marine Control Systems : Guidance, Navigation and Control of Ships, Rigs and Underwater Vehicles*. Trondheim, Norway: Marine Cybernetics AS, 2002, ISBN: 8292356002.
- [44] J. Avila, J. Adamowski, N. Maruyama, F. Takase, and M. Saito, “Modeling and Identification of an Open-frame Underwater Vehicle: The Yaw Motion Dynamics,” *J. Intell. Robot. Syst.*, vol. 66, no. 1–2, pp. 37–56, 2012, DOI:10.1007/s10846-011-9625-x.
- [45] G. Indiveri, “Modelling and Identification of Underwater Robotic Systems,” Thèse de doctorat en ingénierie électronique et informatique, sous la direction du Prof. Ing. R. Zoppoli, University of Genova, Genova, Italy, 1998.
- [46] K. H. Low, “Modelling and parametric study of modular undulating fin rays for fish robots,” *Mech. Mach. Theory*, vol. 44, no. 3, pp. 615–632, Mar. 2009, DOI:10.1016/j.mechmachtheory.2008.11.009.
- [47] X. Liang, J. Zhang, Y. Qin, and H. Yang, “Dynamic Modeling and Computer Simulation for Autonomous Underwater Vehicles with Fins,” *J. Comput.*, vol. 8, no. 4, pp. 1058–1064, 2013, DOI:10.4304/jcp.8.4.1058-1064.
- [48] P. Ridao, A. Tiano, A. El-Fakdi, M. Carreras, and A. Zirilli, “On the identification of non-linear models of unmanned underwater vehicles,” *Control Eng. Pract.*, vol. 12, no. 12 SPEC. ISS., pp. 1483–1499, Dec. 2004, DOI:10.1016/j.conengprac.2004.01.004.
- [49] R. Yang, B. Clement, A. Mansour, H. J. Li, M. Li, and N. L. Wu, “Modeling of a complex-shaped underwater vehicle,” in *2014 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, 2014, no. L, pp. 36–41, Espinho, Portugal, DOI:10.1109/ICARSC.2014.6849759.
- [50] R. Yang, B. Clement, A. Mansour, H. J. Li, and M. Li, “Invited Paper : Robust Control Application To Ciscrea Underwater Vehicle,” in *OCEANS’15/MTS/IEEE*, 2015, Genova, Italy.
- [51] J. Garus, “Optimization of thrust allocation in the propulsion system of an underwater vehicle,” *Int. J. Appl. Math. Comput. Sci.*, vol. 14, no. 4, pp. 461–467, 2004, DOI:10.1.1.385.4601.
- [52] G. H. Golub and C. F. Van Loan, “Matrix Computations,” *Phys. Today*, vol. 10, no. 8, p. 48, 1957, DOI:10.1063/1.3060478.
- [53] R. A. Horn and C. R. Johnson, *Matrix Analysis*, vol. 169. 1985, ISBN: 0521386322.
- [54] T. A. Johansen and T. I. Fossen, “Control allocation—A survey,” *Automatica*, vol. 49, no. 5, pp. 1087–1103, May 2013, DOI:10.1016/j.automatica.2013.01.035.
- [55] E. Omerdic and G. Roberts, “Thruster fault diagnosis and accommodation for open-frame underwater vehicles,” *Control Eng. Pract.*, vol. 12, no. 12 SPEC. ISS., pp. 1575–1598, Dec. 2004, DOI:10.1016/j.conengprac.2003.12.014.
- [56] M. Akmal, M. Yusoff, and M. R. Arshad, “Active Fault Tolerant Control of a Remotely Operated Vehicle Propulsion System,” *Procedia Eng.*, vol. 41, pp. 622–628, 2012, DOI:10.1016/j.proeng.2012.07.221.
- [57] G. Indiveri and G. Parlangeli, “On thruster allocation, fault detection and

- accommodation issues for underwater robotic vehicles.,” in *ISCCSP 2006*, 2006, no. 1, Marrakech, Morocco, DOI:10.1.1.514.79.
- [58] L. Fuqiang, X. Demin, G. Jian, and C. Rongxin, “Fault tolerant control for an autonomous underwater vehicle to dock with thruster redundancy,” in *Chinese Control Conference (CCC)*, 2013, pp. 6186–6190, Xi’an, China.
- [59] M. Chyba, T. Haberkorn, R. N. Smith, S. K. Choi, G. Mariani, and C. M[^]cLeod, “Efficient Control of an Autonomous Underwater Vehicle while accounting for Thruster Failure,” in *International Conference on Computer Applications and Information Technology in the Maritime Industries, COMPIT '08*, 2008, no. 2, Liege, Belgium.
- [60] W. M. Bessa, M. S. Dutra, and E. Kreuzer, “Dynamic Positioning of Underwater Robotic Vehicles with Thruster Dynamics Compensation,” *Int. J. Adv. Robot. Syst.*, vol. 10, p. 1, 2013, DOI:10.5772/56601.
- [61] D. R. Yoerger, J. G. Cooke, and J.-J. E. Slotine, “The influence of thruster dynamics on underwater vehicle behavior and their incorporation into control system design,” *IEEE J. Ocean. Eng.*, vol. 15, no. 3, pp. 167–178, 1990, DOI:10.1109/48.107145.
- [62] L. Pivano, *Thrust Estimation and Control of Marine Propellers in Four- Quadrant Operations*, no. April. NTNU, 2008, ISBN: 9788247162583.
- [63] Jinhyun Kim, Woong Hee Shon, Ho-Gil Lee, and Wan Kyun Chung, “Preliminary thruster control experiments for underwater vehicle positioning,” in *International Conference on Robotics and Automation, 2006. ICRA '06.*, 2006, no. May, pp. 3233–3237, Orlando, USA, DOI:10.1109/ROBOT.2006.1642194.
- [64] N. Mansard, “Enchainement de taches robotiques,” Thèse de doctorat en Informatique, sous la direction de F. Chaumette, Université de Rennes 1, Rennes, France, 2006.
- [65] A. Hanai, H. T. Choi, S. K. Choi, and J. Yuh, “Minimum energy based fine motion control of underwater robots in the presence of thruster nonlinearity,” in *International Conference on Intelligent Robots and Systems, IROS '03*, 2003, vol. 1, pp. 559–564 vol.1, Las Vegas, USA, DOI:10.1109/IROS.2003.1250688.
- [66] A. Hanai, K. Rosa, S. K. Choi, and J. Yuh, “Experimental analysis and implementation of redundant thrusters for underwater robots,” in *International Conference on Intelligent Robots and Systems, IROS '04*, 2004, vol. 2, pp. 1109–1114, Sendai, Japan, DOI:10.1109/IROS.2004.1389545.
- [67] W. M. Bessa, M. . Dutra, and E. Kreuzer, “Thruster dynamics compensation for the positioning of underwater robotic vehicles through a fuzzy sliding mode based approach,” in *ABCMSymposium in Mechatronics*, 2006, vol. 2, no. 1990, pp. 605–612, Ouro Preto, Brazil.
- [68] W. Khalil, E. Dombre, and M. Nagurka, “Modeling, Identification and Control of Robots,” *Appl. Mech. Rev.*, vol. 56, no. 3, p. B37, 2003, DOI:10.1115/1.1566397.
- [69] F. Flacco, A. De Luca, and O. Khatib, “Motion control of redundant robots under joint constraints: Saturation in the Null Space,” in *International Conference on Robotics and Automation, ICRA '12*, 2012, pp. 285–292, Saint Paul, USA, DOI:10.1109/ICRA.2012.6225376.
- [70] Y. Nakamura, H. Hanafusa, and T. Yoshikawa, “Task-Priority Based Redundancy

- Control of Robot Manipulators,” *Int. J. Rob. Res.*, vol. 6, no. 2, pp. 3–15, Jun. 1987, DOI:10.1177/027836498700600201.
- [71] L. Lapiere, “Etude et réalisation de la commande hybride Position/Force d’un robot sous-marin équipé d’un bras manipulateur,” Thèse de doctorat en Génie Informatique, Automatique et traitement du signal, sous la direction de P. DAUCHEZ, Université de Montpellier II, Montpellier, France, 1999.
- [72] A. Liégeois, “Automatic Supervisory Control of the Configuration and Behavior of Multibody Mechanisms,” *IEEE Trans. Syst. Man. Cybern.*, vol. 7, no. 12, pp. 868–871, 1977, DOI:10.1109/TSMC.1977.4309644.
- [73] D. S. Watkins, *Fundamentals of Matrix Computations*. Hoboken, NJ, USA: John Wiley & Sons, Inc., 2002, ISBN: 9780471249719.
- [74] R. Passama, D. Andreu, D. Crestani, and K. Godary-Dejean, “Architectures de controle pour la robotique-Approches et tendances,” *Tech. l’ingénieur*, vol. S7791, p. 21, 2014.
- [75] R. Passama, C. Dony, T. Libourel, and D. Andreu, “Apports d’une approche à composants pour les architectures de contrôle de robots,” in *Conférence Francophone sur les Architectures Logicielles, CAL’06*, 2006, Nantes, France.
- [76] R. Passama, “Conception et développement de contrôleurs de robots. Une méthodologie basée sur les composants logiciels,” Thèse de doctorat en Informatique et Robotique, sous la direction de C. Dony, D. Andreu et T. Libourel, Université de Montpellier II, Montpellier, France, 2006.
- [77] A. G. Raymond, *SOA: Architecture Logique: Principes, structures et bonnes pratiques*. Paris: Softeam, 2011.
- [78] Y. Chen and W. Tsai, *Distributed Service-Oriented Software Development*. USA: Kendall/Hunt, 2008, ISBN: 9780757552731.
- [79] S. Latha and T. Jem, “An Overview of Service-oriented Architecture, Web Services and Grid Computing,” HP, 2005.
- [80] R. J. High, S. Kinder, and S. Graham, “IBM’s SOA Foundation, An Architectural Introduction and Overview,” USA, 2005.
- [81] Y. Chen, A. Sabnis, and M. Garcia-Acosta, “Design and Performance Evaluation of a Service-Oriented Robotics Application,” in *2009 29th IEEE International Conference on Distributed Computing Systems Workshops*, 2009, pp. 292–299, DOI:10.1109/ICDCSW.2009.50.
- [82] V. M. Trifa, C. M. Cianci, and D. Guinard, “Dynamic Control of a Robotic Swarm using a Service-Oriented Architecture,” in *International Symposium on Artificial Life and Robotics, AROB ’08*, 2008, pp. 1–4, Beppu, Japan.
- [83] T. H. Yang and W. P. Lee, “A service-oriented framework for the development of home robots,” *Int. J. Adv. Robot. Syst.*, vol. 10, 2013, DOI:10.5772/55055.
- [84] Y. Chen, A. Sabnis, and M. Garcia-Acosta, “Design and Performance Evaluation of a Service-Oriented Robotics Application,” in *International Conference on Distributed Computing Systems Workshops, ICDCSW ’09*, 2009, pp. 292–299, Montreal, Canada, DOI:10.1109/ICDCSW.2009.50.
- [85] R. Zapata, P. Lépinay, and P. Thompson, “Reactive behaviors of fast mobile robots,” *J.*

Robot. Syst., vol. 11, no. 1, pp. 13–20, 1994, DOI:10.1002/rob.4620110104.

- [86] A. EL JALAOUI, “Gestion Contextuelle de Tâches pour le contrôle d’un véhicule sous-marin autonome,” Thèse de doctorat en Génie Informatique, Automatique et traitement du signal, sous la direction de B. Jouvencel et D. Andreu, Université de montpellier II, Montpellier, France, 2007.
- [87] A. R. Passama, “Rapport LIRMM n°: RR-10026 : Environnement de développement ContrACT,” Montpellier, France, 2010.
- [88] A. Lasbouygues, B. Ropars, R. Passama, D. Andreu, and L. Lapierre, “Atoms Based Control of Mobile Robots with Hardware-In-the-Loop validation,” in *International Conference on Intelligent Robots and Systems , IROS '15*, 2015, Hambourg, Germany.

ANNEXE I : Modules et Services

Cette annexe liste l'ensemble des modules utilisés dans les différents services au sein de l'architecture de contrôle. Pour chaque service, les modules indiqués sont donc composés selon les configurations associées. De plus, chaque module est basé sur un ou plusieurs « atomes » encapsulant les éléments de connaissance impliqués. Ces atomes sont décrits dans la thèse de A. Lasbouygues [40]. Certains modules n'ont pas d'atome associé dans la mesure où ces modules sont assimilables à des drivers (i.e., le contenu du module est étroitement lié à la technologie).

DRIVERS		
Module	Description	Atomes associés
IMU	Centrale inertielle	-
I2C	Communication I2C	-
MTR	Conversion des forces en consignes moteurs	ExpertiseEnginesOperatingRange ExpertiseEnginesParameters ComputePWM CorrectHelixSteps ForceEnginesto EnginesActive
UDP	Communication avec le superviseur	-

DRIVERS EXTENDED		
Module	Description	Atomes associés
I2C	Communication I2C	-
MTE	Conversion des forces en consignes moteurs	ExpertiseEnginesOperatingRange ExpertiseEnginesParameters ComputePWM CorrectHelixSteps ForceEnginesto EnginesActive
UDP	Communication avec le superviseur	-

SONAR		
Module	Description	Atomes associés
SNR	Communication avec le sonar profilométrique	-

DOPPLER VELOCITY LOG		
Module	Description	Atomes associés
DVL	Communication avec le DVL	-

DEPTH CONTROL		
----------------------	--	--

Module	Description	Atomes associés
DPC	Contrôle en profondeur	SubtractorPos SubtractorSpeed ExpertisePID PosePIDAccel
DPD	Estimation de la vitesse	NumericDerivatorPos
DAD	Répartiteur adaptatif	DispatcherAdaptatif
DCP	Répartiteur compressé	DispatcherStaticCompress
DDN	Répartiteur dynamique	DispatcherDynamic
DST	Répartiteur statique	VectorialDispatcher ExpertiseEnginesPositionning

DEPTH CONTROL		
Module	Description	Atomes associés
DEA	Répartiteur adaptatif	DispatcherAdaptatif
DEC	Répartiteur compressé	DispatcherStaticCompress
DED	Répartiteur dynamique	DispatcherDynamic
DES	Répartiteur statique	VectorialDispatcher ExpertiseEnginesPositionning

DRIVERS POST TREATMENTS		
Module	Description	Atomes associés
IPT	Post-traitements des données inertielles	-

DYNAMIC MODEL		
Module	Description	Atomes associés
DMD	Modèle dynamique	DynamicModel ComputeDampings ExpertiseRobot RobotParameters ExpertiseDynamicModel DynamicModelParameters

FRAMESHIFT ACCELERATION		
Module	Description	Atomes associés
FAC	Frameshift accélération	FrameShiftAccel

FRAMESHIFT INVERSE ACCELERATION		
Module	Description	Atomes associés
FIA	Frameshift inverse accélération	FrameShiftInvAccel

FRAMESHIFT INVERSE ANGULAR ACCELERATION		
Module	Description	Atomes associés
FAC	Frameshift inverse accélération Angulaire	FrameShiftInvAngAccel

MAPPING AXIS X ACCELERATION		
Module	Description	Atomes associés
MXA	Mapping accélération (axe X)	-

MAPPING AXIS X FORCE		
Module	Description	Atomes associés
MXF	Mapping force (axe X)	-

MAPPING AXIS X POSITION		
Module	Description	Atomes associés
MXP	Mapping position (axe X)	-

MAPPING AXIS X SPEED		
Module	Description	Atomes associés
MXS	Mapping vitesse (axe X)	-

MAPPING AXIS Y ACCELERATION		
Module	Description	Atomes associés
MYA	Mapping accélération (axe Y)	-

MAPPING AXIS Y FORCE		
Module	Description	Atomes associés
MYF	Mapping force (axe Y)	-

MAPPING AXIS Y POSITION		
Module	Description	Atomes associés
MYP	Mapping position (axe Y)	-

MAPPING AXIS Y SPEED		
Module	Description	Atomes associés
MYS	Mapping vitesse (axe Y)	-

MAPPING AXIS Z ACCELERATION		
Module	Description	Atomes associés

MZA	Mapping accélération (axe Z)	-
-----	------------------------------	---

MAPPING AXIS Z FORCE		
-----------------------------	--	--

Module	Description	Atomes associés
MZF	Mapping force (axe Z)	-

MAPPING AXIS Z POSITION		
--------------------------------	--	--

Module	Description	Atomes associés
MZP	Mapping position (axe Z)	-

MAPPING AXIS Z SPEED		
-----------------------------	--	--

Module	Description	Atomes associés
MZS	Mapping vitesse (axe Z)	-

MAPPING AXIS X ACCELERATION		
------------------------------------	--	--

Module	Description	Atomes associés
MHA	Mapping accélération (axe Phi)	-

MAPPING AXIS PHI FORCE		
-------------------------------	--	--

Module	Description	Atomes associés
MHF	Mapping force (axe Phi)	-

MAPPING AXIS PHI POSITION		
----------------------------------	--	--

Module	Description	Atomes associés
MHP	Mapping position (axe Phi)	-

MAPPING AXIS PHI SPEED		
-------------------------------	--	--

Module	Description	Atomes associés
MHS	Mapping vitesse (axe Phi)	-

MAPPING AXIS THETA ACCELERATION		
--	--	--

Module	Description	Atomes associés
MTA	Mapping accélération (axe Theta)	-

MAPPING AXIS THETA FORCE		
---------------------------------	--	--

Module	Description	Atomes associés
MTF	Mapping force (axe Theta)	-

MAPPING AXIS THETA POSITION		
Module	Description	Atomes associés
MTP	Mapping position (axe Theta)	-

MAPPING AXIS THETA SPEED		
Module	Description	Atomes associés
MTS	Mapping vitesse (axe Theta)	-

MAPPING AXIS PSI ACCELERATION		
Module	Description	Atomes associés
MSA	Mapping accélération (axe Psi)	-

MAPPING AXIS PSI FORCE		
Module	Description	Atomes associés
MSF	Mapping force (axe Psi)	-

MAPPING AXIS PSI POSITION		
Module	Description	Atomes associés
MSP	Mapping position (axe Psi)	-

MAPPING AXIS PSI SPEED		
Module	Description	Atomes associés
MSS	Mapping vitesse (axe Psi)	-

NAVIGATION		
Module	Description	Atomes associés
ICA	Intégration des accélérations	IntegratorTrapAccel EstimateAccel ComputeLinearDamping
ICS	Intégration des vitesses	IntegratorTrapSpeed
JBC	Modèle cinématique	JacobianBodyToCartesian
QTE	Conversion quaternion -> euler	-

PITCH CONTROL		
Module	Description	Atomes associés
PTC	Contrôle du tangage	SubtractorAngle SubtractorAngularSpeed ExpertisePID AnglePIDAngAccel

ROLL CONTROL		
Module	Description	Atomes associés
RLC	Contrôle du roulis	SubtractorAngle SubtractorAngularSpeed ExpertisePID AnglePIDAngAccel

YAW CONTROL		
Module	Description	Atomes associés
YWC	Contrôle du cap	SubtractorAngle SubtractorAngularSpeed ExpertisePID AnglePIDAngAccel

VELOCITY X CONTROL		
Module	Description	Atomes associés
VXC	Contrôle de la vitesse en X	ExpertisePIDSpeed PIDLinearSpeed

VELOCITY Y CONTROL		
Module	Description	Atomes associés
VYC	Contrôle de la vitesse en Y	ExpertisePIDSpeed PIDLinearSpeed

VELOCITY Z CONTROL		
Module	Description	Atomes associés
VZC	Contrôle de la vitesse en Z	ExpertisePIDSpeed PIDLinearSpeed

CENTRING CONTROL		
Module	Description	Atomes associés
FCP	Calcul de force	SonarParameters ExpertiseSonar ExpertiseVirtualProx VirtualProxParameters ExpertiseDVZ DVZParameters ReportDistance CylindricalFrameShiftInv CircularDVZ ResultingForceComputation FrameShiftForce
FZY	Contrôle du centrage en Y	ForceTo0 ExpertiseRobot RobotParameters ExpertiseDynamicModel DynamicModelParameters CriticalDamping DVZKeq ExpertiseVirtualProx VirtualProxParameters ExpertiseDVZ DVZParameters
VPX	Virtual proximeter	CylindricalFrameShift SonarParameters ExpertiseSonar SonarAngles ExtractAngles ExpertiseVirtualProx VirtualProxParameters VirtualProximeter MeasurementValid ExpertiseValidMeas ValidMeasParameters

ANNEXE II : Architecture matérielle embarquée

L'architecture matérielle embarquée au sein du vecteur est structurée autour d'un réseau Ethernet, décrit sur la Figure 2-32II-1. Les Tableau II-1 et Tableau II-2 viennent en complément de cette figure, en indiquant respectivement les adresses IP et le(s) Vlan(s) associé(s) aux équipements.

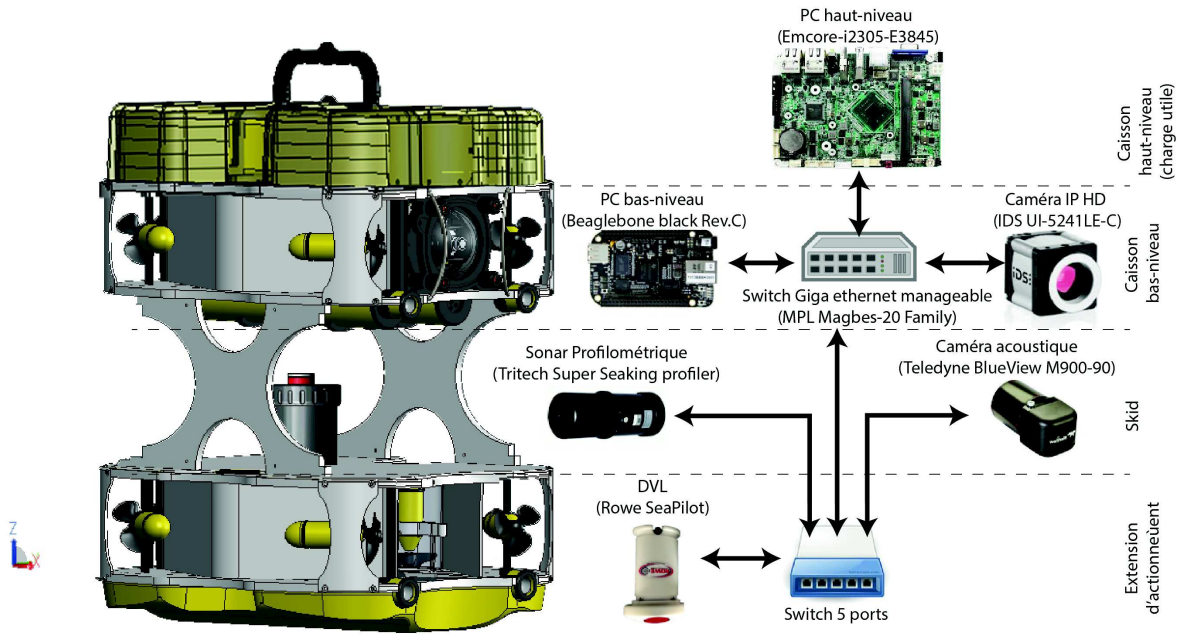


Figure II-1 : Interconnexions des équipements communicants en Ethernet.

Equipement	Adresse IP	Masque	Vlan
Switch manageable	192.168.1.254	255.255.255.0	0, 1, 2, 3
PC bas-niveau	192.168.1.253	255.255.255.0	1, 2
Caméra IP HD	192.168.1.252	255.255.255.0	3
PC haut-niveau	192.168.1.250	255.255.255.0	1, 2, 3
Caméra acoustique	192.168.1.103	255.255.255.0	1
DVL	192.168.1.102	255.255.255.0	1
Sonar profilométrique	192.168.1.101	255.255.255.0	1

Tableau II-1 : Adresse IP et Vlan associés aux équipements

Vlan	Utilisation
0	Réservé pour le système
4095	Non utilisable
1	Vlan par défaut (Vlan capteur métier (sonar, ...))
2	Vlan Contrôle (beagleBone, carte haut niveau, ...)
3	Vlan Vidéo (caméra Ip)

Tableau II-2 : Description des Vlans

Un vecteur Robotique polyvalent pour l'exploration sous-marine faible fond

Depuis maintenant près d'un siècle, des robots sous-marins ont été développés afin de réaliser des tâches spécifiques aux besoins des grands acteurs historiques du domaine (militaires, pétroliers, câbliers ou explorateurs benthiques) sans vraiment se soucier de la polyvalence et de la modularité de la plateforme. L'objectif de cette thèse est d'avoir une réflexion sur une solution technologique et scientifique avec comme domaine applicatif l'environnement faible fond ou confiné. Il s'agit en d'autres termes de concevoir un robot sous-marin que l'on peut faire évoluer aisément tant sur le plan mécanique, électronique qu'informatique. Cet objectif impose de proprement conceptualiser cette « polyvalence » en s'attachant à apporter de l'abstraction dans l'architecture de contrôle que se soit au niveau de l'automatique avec l'expression de la polyvalence liée à l'étage d'actionnement ou de l'informatique avec une architecture basée services, pouvant être composés, afin de répondre à la diversité des besoins applicatifs. L'ensemble de ces travaux a pour point de départ le robot Jack mis au point par l'entreprise Ciscree, partenaire industriel, qui apporte un aspect économique à la nécessité de développer une solution polyvalente pouvant être décliné en une gamme de produits. Ce manuscrit traite de la conception, la réalisation et l'expérimentation de ce vecteur que ce soit en piscine où en environnement réel.

A versatile robotic vector for shallow water exploration

Since almost a century, underwater robots have been developed in order to respond to the specific needs of historical actors of the domain (military, hydrocarbons exploitation, underwater cabling or benthic exploration), without addressing specifically the question of versatility or modularity of the underwater platform. This thesis aims to address these questions on a technological solution dedicated to shallow water or confined environment. In other words, the objective is to realise an underwater system, able to evolve on the mechanical, electronical or software aspects. This requires to properly conceptualise this « versatility » with an abstraction of the control architecture, on the actuation aspect, with the expression of the versatility linked to the actuation systems, or on the software architecture level, with a Service-Oriented-Architecture (SOA) approach, in order to tackle the diversity of the application requirements.

This study is based on the Jack system, developed by the Ciscree Company, which is the industrial partner of this project, and brings the economical aspect as a central requirement. This underlines another view of the versatility question, the development of a range of product for the Ciscree Company.

This thesis proposes the conception, realisation and experimentation of such a versatile underwater system, with test-tank and field validation.